D3.2 - DEMETER Technology Integration Tools - Release 1

Dissemination Level: Public
Submission Date: 9 July 2020

Contents

List of Figures

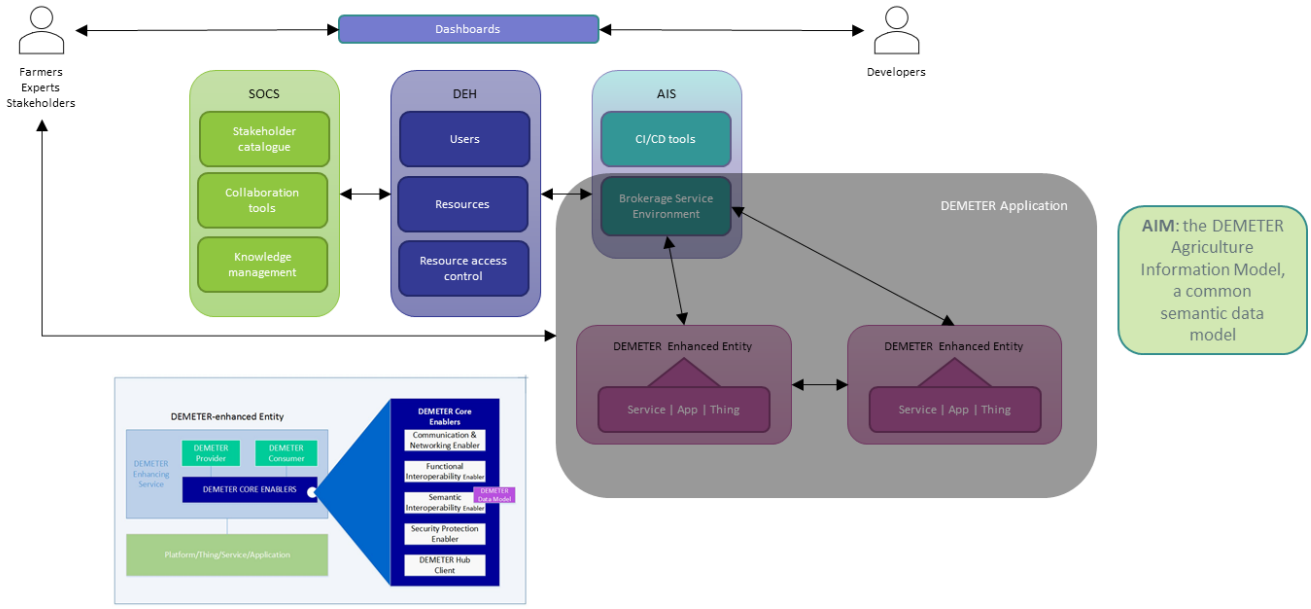
List of Tables

1 Executive Summary

D3.2 is a demonstrator deliverable of the DEMETER project. This document is the accompanying report of this deliverable. D3.2 provides the first release of components and tools that enable solution integration, interoperability with external platforms and deployment support for pilot cases.

The below diagram illustrates the main DEMETER elements to be deployed:

- Stakeholders Open Collaboration Space (SOCS): a knowledge base and co-creation space where farmers, advisors and providers connect.
- DEMETER Enabler HUB (DEH): collects all the resources that are available to be used by a solution and enables access to them.
- Agricultural Interoperability Space (AIS): provides interoperability mechanisms to develop and deploy a solution.
- Dashboards: sole entry points to the DEMETER ecosystem.
- DEMETER-enhanced Entity (DEE): A Service, Application, Platform, or Thing wrapped with DEMETER enabler functionalities to act as a DEMETER consumer and/or producer. Many of these DEEs interoperate with each other to form an application solution.
- Agriculture Information Model (AIM): a common semantic data model to be used for the information exchange.



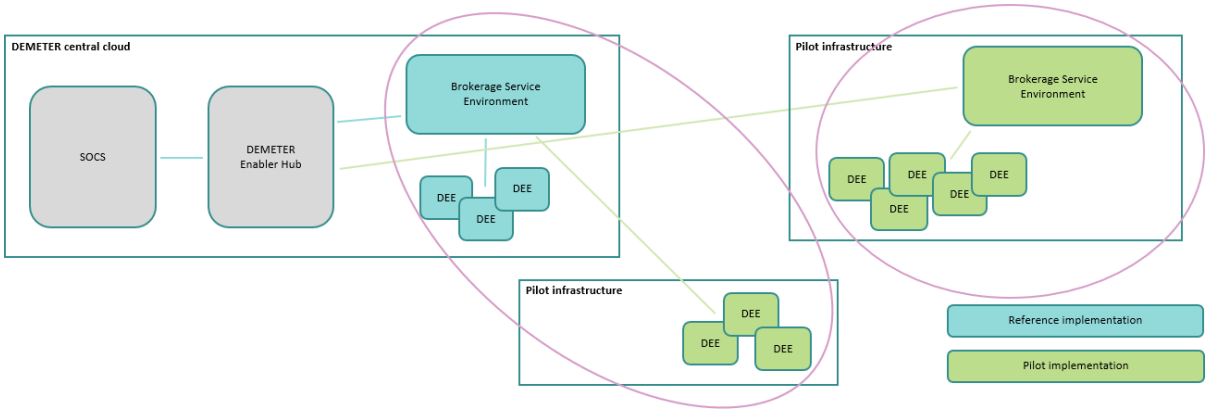
For each of the below components this document provides description, multiple architectural views, interface definition and notable details about the used technologies and the implementation:

- Brokerage Service Environment: a microservices-based environment used to facilitate the registration, discovery, and communication of the DEEs.
- Access Control Server: offers authentication, authorisation, traceability functionalities to the brokerage environment.
- DEMETER Enabler HUB (DEH): collects all the resources that are available to be used by a solution and enables access to them.
- Core Enablers for integration: specifications for core enablers that need to be implemented by DEEs in order to interoperate in a DEMETER application.

Finally, the DEMETER CI/CD tools and the Verification & Validation plan are presented.

For more information on AIM see deliverables D2.1 and D2.2. For more information on SOCS see D4.2. D3.1 “DEMETER reference architecture - Release 1” provides overview and further information for all DEMETER components.

The deployment of the above components and their use in the DEMETER Pilots are depicted in the below diagram. Pilots can either user their infrastructure and deploy there (eclipse on the right) or they can rely on components deployed in the DEMETER's central cloud and use their infrastructure to add extra DEEs (eclipse in the centre).



This scheme enables providing a concrete implementation to be used by the pilot applications and guide further development, while offering full flexibility for the application configuration and deployment to facilitate the highly different pilot needs and various business models.

2 Acronyms

ACS	Access Control Server
AIS	Agricultural Interoperability Space
API	Application Programming Interface
BID	Business Intelligence Dashboard tool
BS	Brokerage Server
BSE	Brokerage Service Environment
CI/CD	Continuous Integration / Continuous Deployment
CoAP	Constraint Application Protocol
CRUD	Create Read Update Delete
DAE	DEMETER Advanced Enabler
DAO	Data Access Object
DEE	DEMETER-enhanced Entity
DEH	DEMETER Enabler HUB
DSS	Decision Support System
DTLS	Datagram Transport Layer Security
ETSI	European Telecommunications Standards Institute
FMIS	Farm Management Information System
GA	Grant Agreement
GDPR	General Data Policy Regulations
GE	Generic Enablers
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IdM	Identity Management
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IOT	Internet of Things
IP	Internet Protocol
ISO	International Organization for Standardisation
IT	Information Technology
JSON	Java Script Object Notation
KPI	key Performance Indicator
LAN	Local Area Network
MQTT	Message Queuing Telemetry Transport
NGSI	Next Generation Sensors Initiative
NGSI-LD	Next Generation Sensors Initiative - Linked Data
OneM2M	One Machine to Machine
PAN	Personal Area Network
PDP	Policy Decision Point
PEP	Policy Enforcement Point
RDF	Resource Description Framework
REST	Representational State Transfer
RFID	Radio Frequency Identification Device
RPC	Remote Procedure Calls
RTPS	Real Time Publish Subscribe
SaaS	Software as a Service

SDK	Software Development Kit
SOCS	Stakeholders Open Collaboration Space
SQL	Structured Query Language
SSL	Secure Sockets Layer
SR	Service Registry
TBD	To Be Determined
TDD	Test Driven Development
TLS	Transport Layer Security
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
WAN	Wide Area Network
WSN	Wireless Sensor Network
XACML	Extensible Access Control Markup Language
XML	Extensible Markup Language

3 List of Authors and Reviewers

Organization	Author
INTRA	Ioannis Oikonomidis, Athanasios Poulakidas, Dimitrios Skias
DNET	Srdjan Krco, Nenad Gligoric
ENG	Antonio Caruso, Maria Francesca Cantore
VICOM	Oscar Miguel Raul Orduna, Loinaz Merino
UMU	Manuel Mora González
ODINS	Juan Antonio Martinez
ICCS	Ioannis Vetsikas, Georgios Routis, Marios Paraskevopoulos, Ioanna Roussaki
TECNALIA	Sonia Bilbao, Belén Martínez, Fernando Jorge
LESP	Karel Charvát jr, Michal Kepka
ROTECH	Lorenzo Bortoloni, Damiano Vallocchia Nadia, Caterina Zullo Lasala
SIVECO	Mihai Angheloiu
ATOS	Sergio Salmeron

Organization	Reviewer
ATOS	Jesús Benedicto
ICE	John Beattie, Oscar Garcia Perales

4 Introduction

D3.2 is a demonstrator deliverable of the DEMETER project. This document is part of this deliverable. D3.2 provides the first release of components and tools that enable solution integration, interoperability with external platforms and deployment support for pilot cases. The following Tasks contributed to D3.2: T3.2, T3.3, T3.4, T3.5, T3.6.

Figure 1 illustrates the main DEMETER elements to be deployed:

- Stakeholders Open Collaboration Space (SOCS): a knowledge base and co-creation space where farmers, advisors and providers connect.
- DEMETER Enabler HUB (DEH): collects all the resources that are available to be used by a solution and enables access to them.
- Agricultural Interoperability Space (AIS): provides interoperability mechanisms to develop and deploy a solution.
- Dashboards: sole entry points to the DEMETER ecosystem.
- DEMETER-enhanced Entity (DEE): A service, application, platform or thing wrapped with DEMETER enabler functionalities to act as a DEMETER consumer and/or producer. Many of these DEEs interoperate with each other to form an application solution.
- Agriculture Information Model (AIM): a common semantic data model to be used for the information exchange.

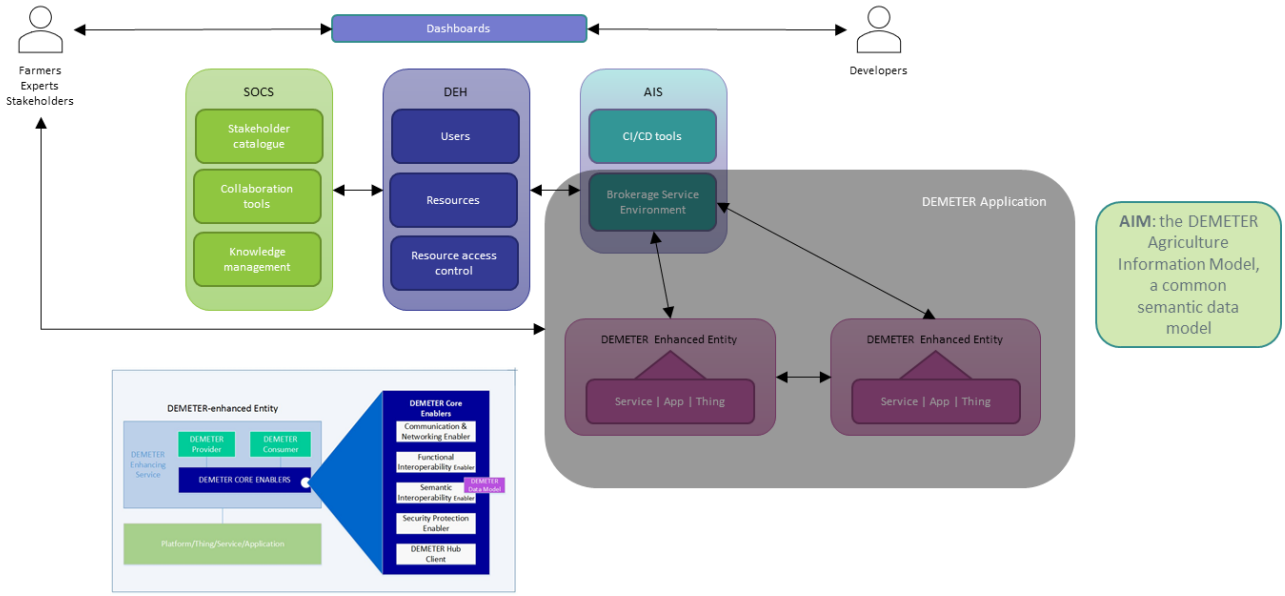


Figure 1: DEMETER elements

The remaining of this document is comprised of the following sections:

Section 5 provides the overall architecture of the DEMETER reference implementation. It also provides an overview of its collected requirements and their mapping to the implementation components.

Section 6 presents the Brokerage Service Environment, a microservices-based environment used to facilitate the registration, discovery, and communication of the DEEs.

Section 7 presents the Access Control Server, which offers authentication, authorisation, traceability functionalities to the brokerage environment.

Section 8 presents the DEMETER Enabler HUB (DEH) which offers all available Enablers in a catalogue for users.

Section 9 specifies the core Enablers for integration. These enablers need to be implemented by DEEs in order to interoperate in a DEMETER application.

Section 10 describes the CI/CD tools and how they are deployed to assist in integrating and deploying a DEMETER application.

Section 11 presents the Verification and Validation plan to be used for the implementation of the DEMETER applications.

Section 12 provides some conclusions and next steps.

Appendix A lists the full details of the requirements whose overview was provided in Chapter 5.

Appendix B provides the template used to specify each DEMETER Enabler.

Appendix C presents the survey and its results that guided the development of the DEH.

Appendix D provides the template used to represent the general information of any DEMETER component.

5 Architecture of the Reference Implementation

Based on the DEMETER's reference architecture presented in the deliverable D3.1, this deliverable moves a step forward and illustrates the Architecture of the Reference Implementation.

5.1 *Architecture (Physical view, Process view)*

This section depicts the Physical and Process View of the DEMETER Reference Implementation. More specifically, Figure 2 describes the 3 major components of DEMETER platform, namely Stakeholder Open Collaboration Space, DEMETER Enabler HUB and Brokerage Service environment. Included in these three major components lies, the Security component (presented in section 7 as the Access Control Server component). In addition, it illustrates the interoperation activities between DEMETER Enhanced Entities (DEE) and DEMETER's Reference Implementation. DEE consists of a set of either an app, a service, or a device along with a set of core Enablers and Advanced Enablers.

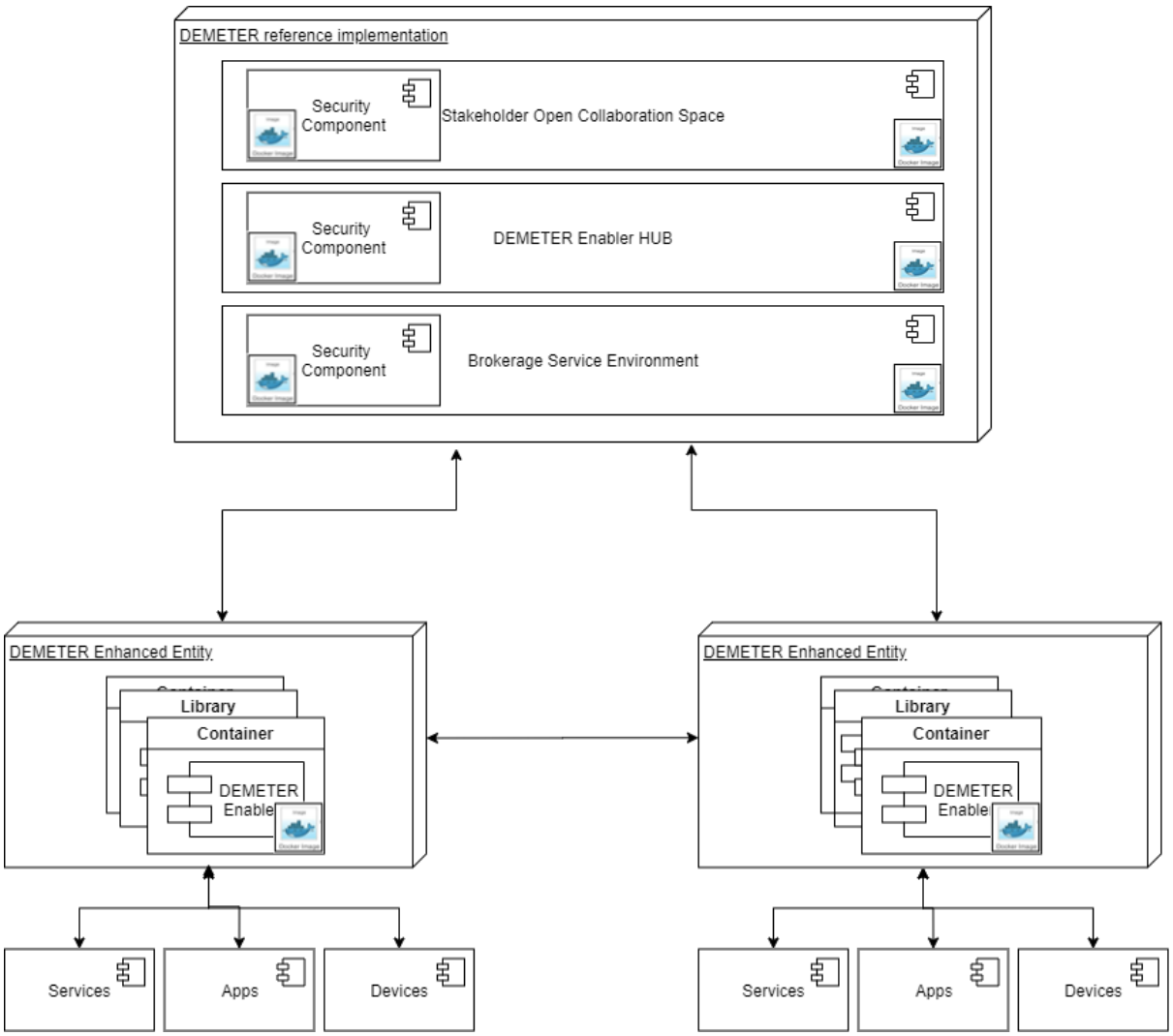


Figure 2: Reference Implementation Deployment diagram

Figure 3 depicts the Pilots' deployment schema. DEMETER's Pilots can either user their infrastructure and deploy in their premises the DEMETER Brokerage Service Environment and the DEMETER Enhanced Entities (eclipse on the right) or they can rely on the BSE that is deployed in the DEMETER's

Central Cloud and use this infrastructure in order to enable the communication of their DEE (eclipse in the centre). In the same figure, the box on the left depicts the DEMETER Central Cloud where the BSE, the SOCS, the DEH and DEE resides.

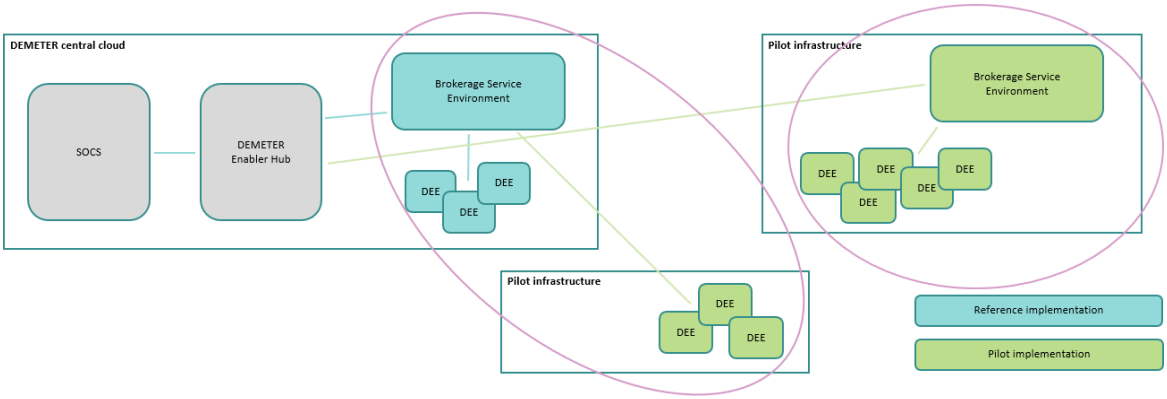


Figure 3: Pilots Deployment diagram

Figure 4 and Figure 5 present sequence diagrams that illustrates a set of processes that are offered by the DEMETER platform. The main actors of the diagrams are the DEMETER Provider who "provide" some resource to the DEMETER platform, the DEMETER platform which comprises the Security component, the BSE, the DEH and the SOCS, and finally the DEMETER Consumer who "consumes" some resource that is offered by the DEMETER platform.

Figure 4 illustrates the resource registration process that a provider initiates to register the resource in DEMETER's catalogue. DEMETER's security component facilitates the Authentication and Authorization process, subsequently the provider registers the resource to the BSE. Once the registration is confirmed the provider can register the resource with the DEH.

Figure 5 illustrates the process of resource discovery from the side of the consumer. Firstly, DEMETER consumer logs in to the DEMETER platform through the Security component. Then, through the SOCS environment, the consumer can investigate a solution that matches their needs. Subsequently, the consumer browses on the DEH to find the right Enablers that would facilitate the access towards the resource that needs to consume. Once those Enablers are deployed, he discovers the relevant resource via the BSE. The BSE returns to the consumer the access information for that specific resource.

The final part of the process described depicts the consumer, possessing the access information that was given to him by the BSE, finding the requested resource and proceed in making requests and receiving the subsequent responses.

Figure 6 and Figure 7 complement the sequence diagrams described above and present the activity diagrams of the process described above.

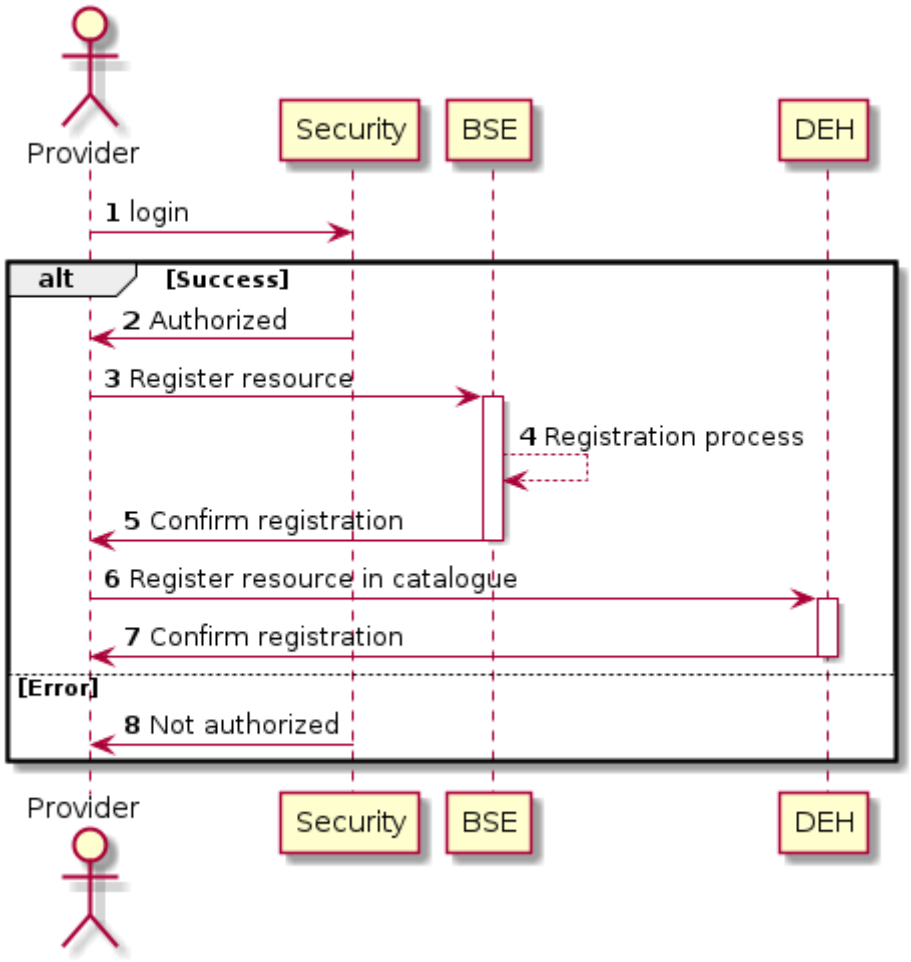


Figure 4: Sequence diagram - Provider

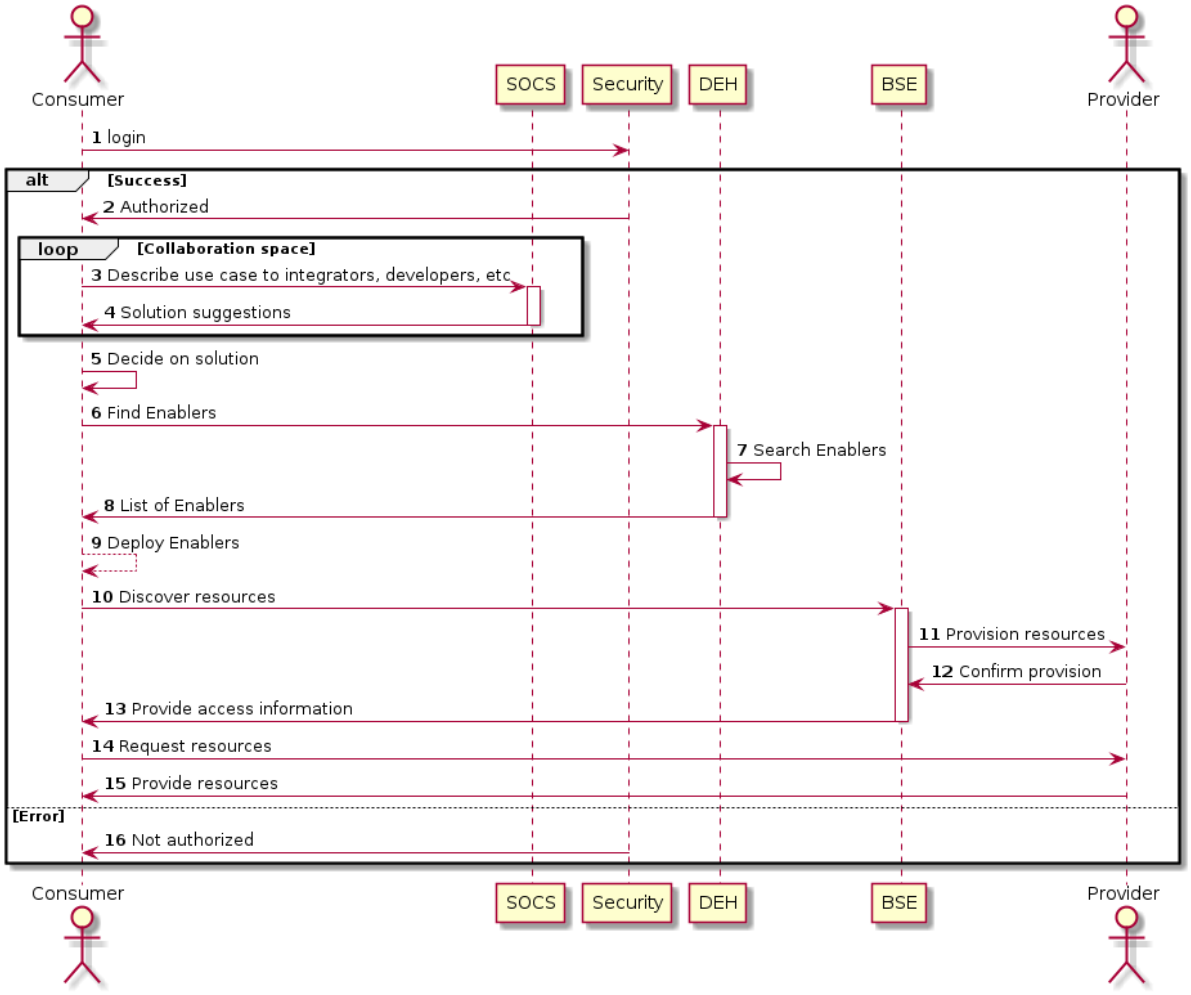


Figure 5: Sequence diagram - Consumer

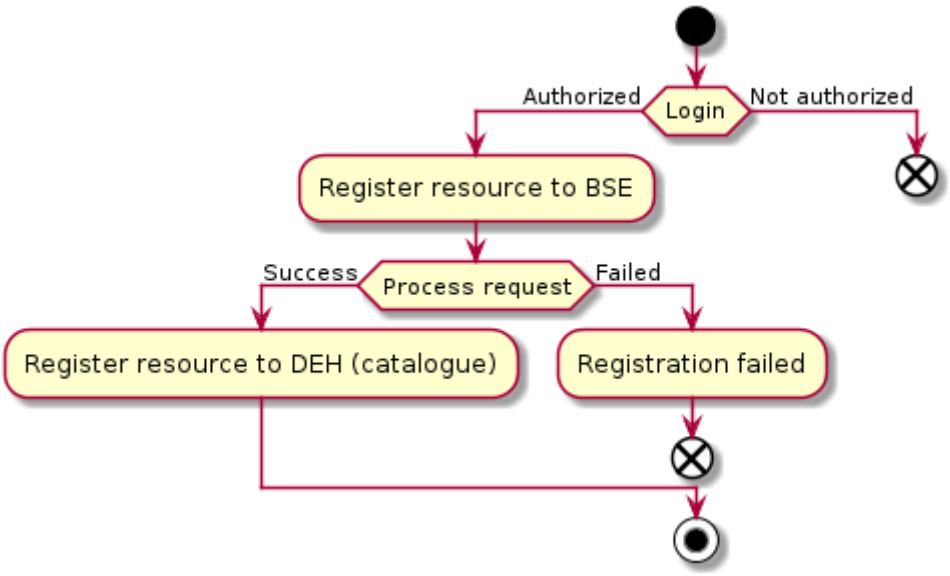


Figure 6: Activity diagram - Provider

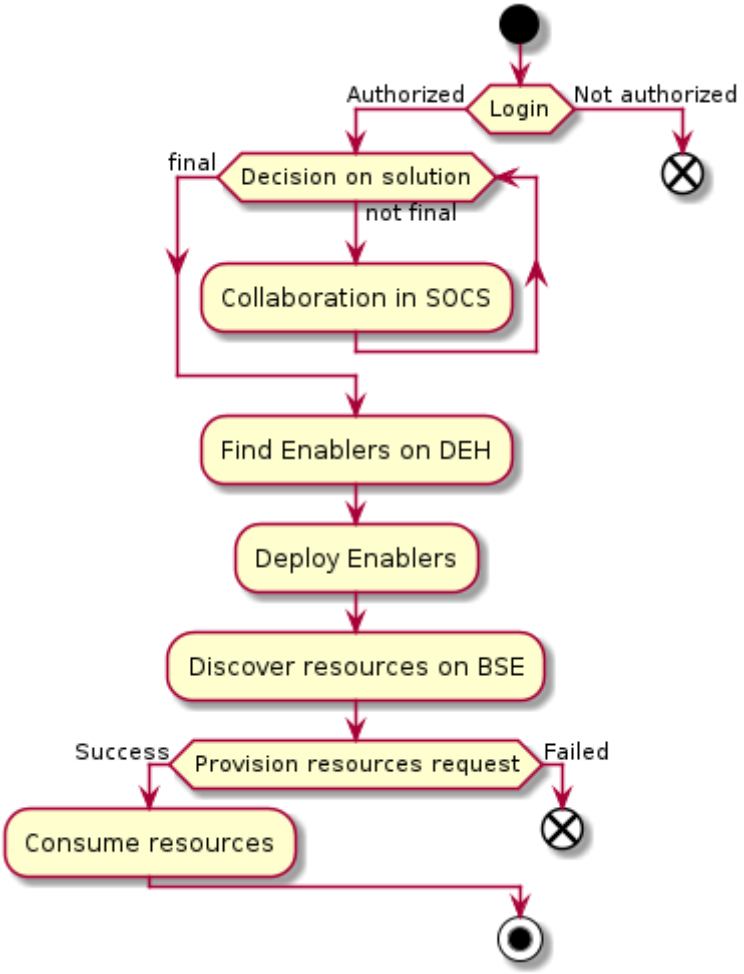


Figure 7: Activity diagram - Consumer

5.2 *Requirements Mapping*

Table 1 below summarizes the functional and non-functional requirements that refer to the Reference Implementation.

Table 1: Summary of Functional and Non-functional requirements for Reference Implementation

ID	Name	Related Component
TI1.1	Utilization of existing standards	Enablers
TI1.2	Support of Communication Protocol Standards	Enablers
TI1.3	Support of Geospatial Interoperability Standards	Enablers
TI1.4	Provide interoperability with existing cloud platforms	Enablers
TI1.5	HTTP REST API(s)	Enablers, BSE
TI1.6	Pub/sub and messaging queue mechanisms	Enablers, BSE
TI1.7	Compliance with system domain standards	Enablers, BSE, DEH, Security
TI1.8	Data formats	Enablers, BSE, DEH
TI2.1	Service description definition	Enablers, BSE, DEH
TI2.2	Services provisioning maintaining data security and privacy	Enablers, BSE, DEH, Security
TI2.3	Services registration to DEMETER Enabler Hub	Enablers, BSE, DEH
TI2.4	Services' categorization	DEH
TI3.1	Secure transport layer (TLS, SSH, etc.)	Enablers, Security
TI3.2	GDPR technical requirements	Enablers, BSE, DEH, Security
TI3.3	Combination of physical/wireless communications and Internet backbone networks	Enablers
TI3.4	Control devices sharing information	Enablers
TI4.1	Attribute Based Access Control or Distributed Capabilities Access Control component	Enablers, Security
TI4.2	Authentication and authorization mechanisms for services, accessing resources and information audit tools	Enablers, Security
TI4.3	Data protection and privacy on software execution, network communications and integrated solution security	Enablers, Security, BSE, DEH
TI4.4	Identity management, access control and audit log	Enablers, Security
TI4.5	Encrypted communications, integrity controls and electronic signature functionalities	Enablers
TI5.1	Data storage systems access management	Enablers
TI5.2	Registration the capabilities of a resource	Enablers, DEH, BSE
TI5.3	Multiple devices bulk operations	Enablers
TI5.4	Resource/device sharing rules	Enablers, DEH, BSE
TI6.1	DEMETER Enablers deployment	Enablers
TI6.2	DEMETER Enablers compliance	Enablers, BSE, DEH
TI6.3	DEMETER deployment tests	Enablers, BSE
TI6.4	DEMETER runtime environment agnostic	Enablers
TI6.5	Deployment process documentation	Enablers, DEH
TI6.6	Deployment software life-cycle management	Enablers, DEH
TI6.7	Deployment process security	Enablers

TI7.1	Service/application life-cycle management methodology	Enablers, DEH, BSE, Security
TI7.2	Technical requirements review	Enablers, DEH, BSE, Security
TI7.3	Components' testing	Enablers, DEH, BSE, Security
TI7.4	Development teams' communication	Enablers, DEH, BSE, Security
TI7.5	Component maintenance	Enablers, DEH, BSE, Security
TI7.6	Service/application life-cycle management software suites	Enablers, DEH, BSE, Security
TI8.1	CRUD to HTTP methods mapping	Enablers, DEH, BSE, Security
TI8.2	Proper HTTP response codes	Enablers, DEH, BSE, Security
TI8.3	Searching, sorting, filtering, and pagination	Enablers, DEH, BSE, Security
TI8.4	Stateless Authentication & Authorization	Enablers, DEH, BSE, Security
TI8.5	Usage of Swagger for Documentation	Enablers, DEH, BSE, Security
TI8.6	REST-based services	Enablers, DEH, BSE, Security
TI8.7	Access control mechanisms in API(s)	Enablers, DEH, BSE, Security
TI8.8	API and application documentation	Enablers, DEH, BSE, Security
TI9.1	Semantic resource registry	DEH
TI9.2	Discovery Management	DEH
TI9.3	Query Management	DEH
TI9.4	Rate services in publish & subscribe mechanism	DEH
TI9.5	Resource Access Control	DEH
TI9.6	Query Management	DEH
TI9.7	Publish & Subscribe Notification	DEH
TI9.8	Enablers Information Management	DEH
TI9.9	DEH Scalability & Availability	DEH
TI9.10	Licensing	DEH
TI9.11	Data encryption in communications	DEH
TI9.12	Service User Advisory	DEH
TI9.13	Accounting Management	DEH
TI9.14	Semantic Interoperability Framework	DEH
TI9.15	Application portability	DEH
TI9.16	System security services	DEH
TI9.17	System availability	DEH
TI9.18	External registration and provisioning	DEH
TI9.19	Data synchronization	DEH
TI9.20	Data federation	DEH
TI9.21	Technology specification	DEH

TI9.22	DEH modules characteristic definition	DEH
TI9.23	Data management	DEH
TI9.24	Data fusion	DEH
TI9.25	Monitoring & Audit	DEH
TI9.26	Information Management	DEH
TI9.27	Data Semantic Interoperability	DEH
TI9.28	Data Resource Definition	DEH
TI9.29	Resource Management (CRUD operations)	DEH
TI9.30	Web service interoperability	DEH
TI9.31	Resource compatibility checker	DEH
TI9.32	Agriculture interoperability space resources	DEH
TI9.33	Data Discovery Management	DEH
TI9.34	Rating service	DEH
TI9.35	Resource statistics report	DEH
TI9.36	Collection of enablers system	DEH
TI9.37	User profile management	DEH
TI9.38	Responsive web GUI	DEH
TI9.39	User account management	DEH
TI9.40	User private home page	DEH
TI9.41	User registration web page	DEH
TI9.42	Resources Management web page	DEH
TI9.43	Interoperability marketplace and catalogues solution	DEH
TI9.44	DEH solutions web page	DEH
TI9.45	Team services	DEH
TI10.1	Stakeholder access	DEH, Security
TI10.2	Account management roles functionality	Enablers, DEH, BSE, Security
TI10.3	Distinguishing a) internal and external stakeholders and b) primary and secondary stakeholders	DEH, Security
TI10.4	Stakeholders' categorization	Enablers, DEH, BSE, Security
TI11.1	Feedback from end-users	DEH
TI11.2	Upvoting mechanism	DEH
GNFR.1	Business analytic data visualization suite	Enablers
GNFR.2	Decision Support System Dashboards	DEH
GNFR.3	Web applications usability	DEH
GNFR.4	Web application stylesheet	DEH
GNFR.5	Web application friendliness	DEH
GNFR.6	Business analytic data visualization suite	Enablers
GNFR.7	DSS dashboard outcomes data visualization	Enablers
GNFR.8	DSS dashboard notification	Enablers
GNFR.9	DSS Dashboard widget	Enablers

6 Brokerage Service Environment

6.1 *Description*

The Brokerage Service Environment (BSE) is a core component of DEMETER architecture, which facilitates the registration, discovery and ultimately communication process for the DEMETER-enabled resources in a secure and privacy preserving manner. In the framework of DEMETER, a resource coupled with the necessary enablers (core and advanced) is named DEMETER enhanced entity (DEE). A DEE once authenticated and authorized by the BSE can register as a service with the BSE specific registry. Subsequently, it becomes discoverable by all the other registered DEEs. Finally, based on the suitable core and advanced enablers that each DEE implement and after resource provisioning from the BSE, DEEs should be able to communicate directly between each other. In addition to the functionalities, BSE can interconnect (interface) with DEMETER HUB and facilitate the registration process of DEEs that are governed by the BSE to the HUB.

The BSE will be implemented as a self-contained application that would enable an external party to deploy it as a complete brokerage service solution. The BSE accompanied by a publish-subscribe communication mechanism that addresses the required communication data throughput and fits the specific needs of that external party (e.g. RabbitMQ, KAFKA etc.) realize the backbone of the DEMETER reference architecture.

The following sections describe BSE's core components and their interactions, along with the sequence diagrams that illustrate the data flow between them.

6.2 *Development View*

The development view illustrates a system from a programmer's perspective and is also known as the implementation view. It uses the UML Component diagram to describe BSE components.

6.2.1 **Component diagram**

Figure 8 below illustrates the major components of the BSE. Its core components are the Access Control Server (ACS), the Brokerage Server (BS) and the Service Registry (SR). The ACS provides for the authentication and the authorization of the DEEs that request to be included in the BSE, The BS realizes the DEE registration, discovery, and the provisioning functionality.

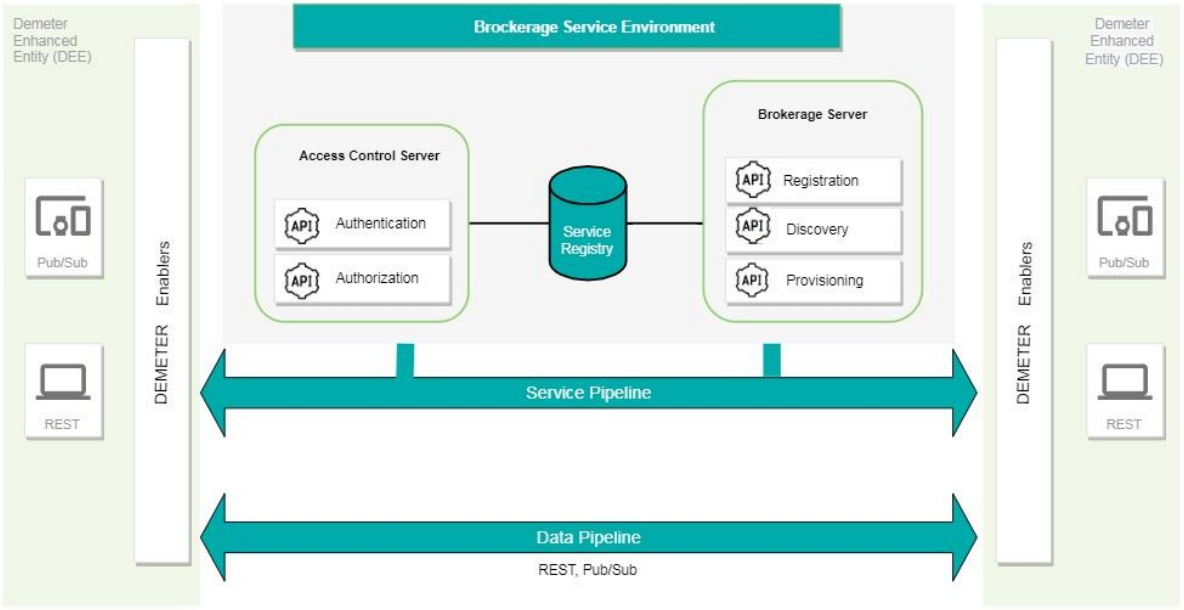


Figure 8: BSE component diagram

6.2.2 **Building blocks (components)**

6.2.2.1 Access Control Server

Access Control Server and its sub-components are described in detail in section 7. BSE is utilizing the functionality provided by this component.

6.2.2.2 Service Registry

In the context of Brokerage Service Environment (BSE), the Service Registry implements a RESTful interface through which it communicates on one hand with Access Control Server (ACS) and on the other with Brokerage Server (BS). Service Registry is used to store user and service related meta data in a persistent manner. More specifically, it holds, user authentication credential information (where necessary) and access tokens that are generated by the ACS and are used from third party services to access and interoperate with BSE endpoints. Furthermore, it stores service-related meta-data that is required or generated by the Brokerage Server (BS)

6.2.2.3 Brokerage Server

In the context of Brokerage Service Environment (BSE), the Brokerage Server (BS) purpose is to facilitate the registration, discovery, and provisioning service. It is envisioned to be built on top of Consul (https://www.consul.io/) which is a service mesh solution providing a full featured control plane with service discovery, configuration, and segmentation functionality. Through Brokerage Server (BS) and the RESTful interface that it implements, a third-party service can get registered, discovered, and queried through the BSE. The BS, interfaces with Service Registry where it stores services' meta-data in a persistent manner. In addition, where it is necessary from a security or administrative point of view, the BS incorporates Access Control Lists through tokens that can be used to confine each service discovery environment.

6.3 *Process View*

Figure 9 illustrates a Brokerage Service Environment (BSE) sequence diagram that depicts an overview of the core functionalities provided by the BSE. Each functionality is presented in its own frame where the data flow is described.

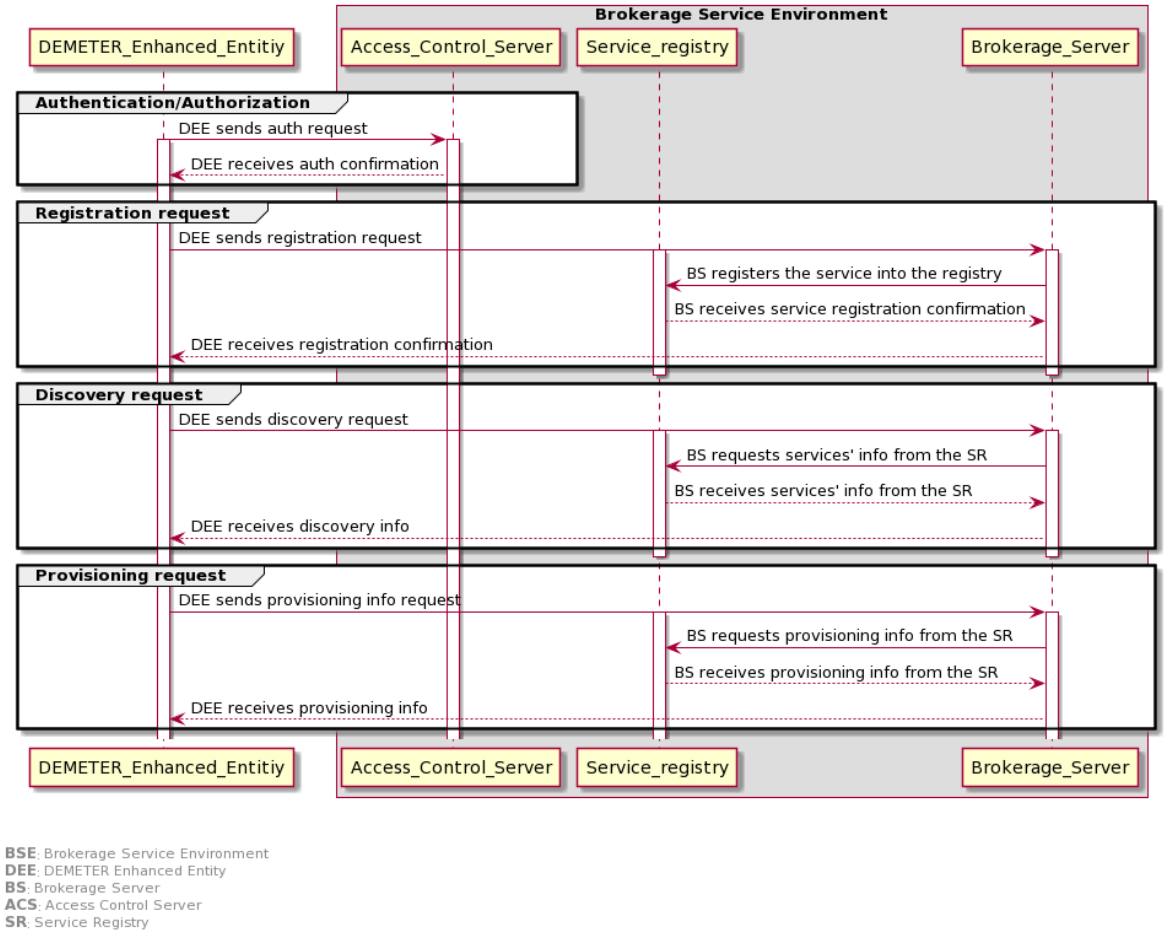


Figure 9: BSE sequence diagram

6.4 *Interfaces*

6.4.1 Data Models used in interfaces

Name	BSE data model	
Property	Type	Description
timestamp	Timestamp	The transaction timestamp
resource_id	String	The resource unique id
resource_name	String	The resource name
resource_access_info	JSON	Information on how to access the resource (e.g., port, protocol, URL, etc)
resource_metadata	JSON	Metadata information for the resource (e.g., vendor, version, etc)
resource_validation_info	JSON	Information on how to validate the resource (e.g., validation

		endpoints, expected responses, etc)
resource_dependencies	Array	Dependencies on other resources
resource_usage_info	JSON	Information on the usage of the resource (e.g., accepted request rate, restrictions on concurrent consumers, etc)
resource_tags	Array	Tags for discoverability
start_time	Timestamp	Start time (e.g., the start time in a resource provisioning request)
end_time	Timestamp	End time (e.g., the end time in a resource provisioning request)
user_id	String	The provider/consumer unique identifier
provision_request_info	JSON	Information on the resource provisioning request (e.g., requested duration, rate, number of devices, number of users, etc)
provision_access_info	JSON	Information on the provisioning (e.g., duration of access, rate of access, restrictions on concurrent connections, etc)

6.4.2 Description of APIs

Title	Register resource to BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://brokerage/api/v1/resource	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
timestamp	The timestamp of registration
user_id	The unique identifier of the provider
resource_name	The name of the resource to be registered
resource_access_info	The access info of the resource
resource_metadata	The metadata of the resource
resource_validation_info	The validation info of the resource

resource_dependencies	The dependencies of the resource
resource_usage_info	The usage information of the resource
resource_tags	The tags for the resource
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {resource_id}	Request was successful
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Modify registered resource to BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://brokerage/api/v1/resource	
Method This field holds the type of the Method used	
PUT	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identified of the provided
resource_id	The unique identifier of the resource
Optional:	
resource_name	The name of the resource
resource_access_info	The access info of the resource
resource_metadata	The metadata of the resource
resource_validation_info	The validation info of the resource
resource_dependencies	The dependencies of the resource
resource_usage_info	The usage information of the resource
resource_tags	The tags for the resource
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: { }	Resource was successfully modified

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Remove registered resource from BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://brokerage/api/v1/resource	
Method This field holds the type of the Method used	
DELETE	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the provider
resource_id	The unique identifier of the resource
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: { }	Resource was successfully deleted
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Discover registered resource from BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	

http://brokerage/api/v1/resource	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the consumer
Optional:	
resource_id	The unique identifier of the resource
resource_name	The name of the resource
resource_metadata	The metadata of the resource
resource_tags	The tags for the resource
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: [resource_id: { resource_name: String, resource_metadata: JSON, resource_validation_info: JSON, resource_dependencies: [String], resource_usage_info: JSON, resource_tags: [String] }]	An array of resource objects discovered
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Provision registered resource
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://brokerage/api/v1/provision	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request

Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the consumer
resource_id	The unique identifier of the resource
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {resource_access_info}	Provisioning and access information
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Check compatibility of resource
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://brokerage/api/v1/compatibility	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the consumer
resource_id	The unique identifier of the resource
Optional:	
resource_validation_info	The validation info of the resource
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: { }	Compatibility check info
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found

403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

6.5 *Technologies and implementation details*

The Brokerage Service Environment (BSE) will be implemented in Django Framework (Python-based framework, https://www.django-rest-framework.org/)

and will be realized as a containerized application with a self-contained execution environment. It consists of a set of Docker containers that hold the Brokerage Server (BS) and the Access Control Server (ACS). In addition, BSE also implements a REST API which is based on the Django Rest Framework. The Brokerage server will be developed based on the Consul service discovery and configuration system.

7 Access Control Server

7.1 *Description*

The security components provide will provide the following three functionalities to other DEMETER components and pilots implementations:

- Authentication
- Authorisation
- Traceability

These functionalities have been implemented in six main security components: Identity Manager, XACML PDP, Capability Manager, PEP Proxy, Traceability Agent and Traceability blockchain repository. These components expose methods using a REST API as described in the following sub sections.

7.2 Development View

7.2.1 **Component diagram**

The following diagram depicts the security components and their relationships in order to provide the authentication, authorisation and traceability functionalities:

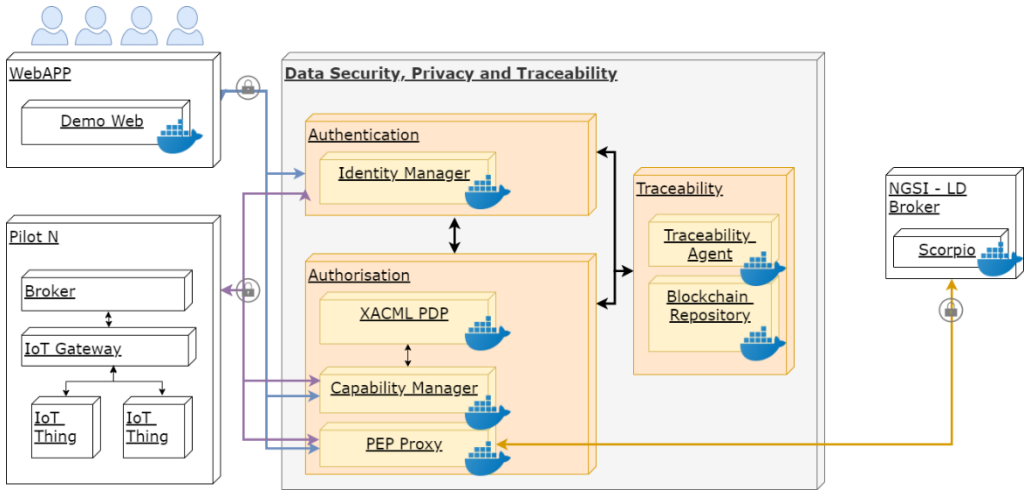


Figure 10: Security component diagram

The authentication functionalities will be provided by the FIWARE Identity Manager component. The authorisation functionalities will be provided by the DCabBAC module, which contains three subcomponents: XACML PDP, Capability Manager and PEP Proxy. The traceability functionalities will be provided by the Traceability Agent, which will log the use of the authentication/authorisation token within the traceability blockchain repository.

These building blocks are described in the following sub section.

7.2.2 Building blocks (components)

Description of the Data Security Components:

- Identity Manager
- XACML PDP
- Capability Manager
- PEP Proxy

- Traceability Agent (VICOM)
- Traceability Blockchain Repository (VICOM)

7.2.2.1 Identity Manager

The Demeter Identity Manager (IdM) component is based on the FIWARE Keyrock GE (https://fiware-idm.readthedocs.io/en/7.4.0/) and will provide the Keyrock's REST API for authentication based on the OAuth 2.0 protocol. The OAuth 2.0 protocol supports several grants ("methods") types for a client application to acquire an access token (which represents a user´s permission for the client to access their data) which can be used to authenticate a request to the Keyrock API endpoint. The following methods for authentication are provided:

- **Authorization Code**: defined for apps running on a web server, where the user will be redirected to the Keyrock server.
- **Username and Password**: for logging in with a username and password directly in the web server.
- **Client credential**: suitable for machine-to-machine authentication where specific user´s permission to access data is not required
- **Refresh token**: to refresh the authentication token before its expiration time.

7.2.2.2 XACML PDP

The XACML PDP manages the access control policies and decides who can access to a resource and what actions can perform over that resource.

The PDP (*Policy Decision Point*) evaluates XACML (*eXtensible Access Control Markup Language*) policies in XML representation. With the specified policies, a request from the Capability Manager made to the PDP, that has the location of the policies, is evaluated to decide if the access or action in the request can be performed or not sending back a response. This communication is sent encoded in JSON, which provides a less verbose representation of the information and improves the request processing as well. The next text shows an example of an XACML policy in XML format:

```
<Policy PolicyId="example">
  <Rule Effect="Permit" RuleId="001">
    <Target>
      <Subject>
        <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Peter</AttributeValue>
        </SubjectMatch>
      </Subject>

      <Resource>
        <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
```

```
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">https://215.64.19.203:1020/ngsi-
ld/v1/entities?type=http://www.w3.org/ns/sosa/Sensor;idPattern=urn:ngsi-ld:Sensor:temperature.*
        </AttributeValue>
            </ResourceMatch>
          </Resource>

          <Action>
            <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">GET</AttributeValue>
            </ActionMatch>
          </Action>
        </Target>
    </Rule>
</Policy>
```

The PDP is deployed as a Web Service to be accessed by the authorization entity acting as Policy Enforcement Point (PEP) through the exchange of HTTP messages with JSON payloads containing the XACML requests or responses. The PDP is based on Web technologies to be a scalable and lightweight solution so it can be applied to any large-scale deployment that requires XACML as policy language. XACML PDP achieves clear performance improvements over other existing solutions in terms of scalability and efficiency.

The next figure shows a flow chart with an XACML PDP in an authorization process example:

1. The Capability Manager asks the XACML PDP sending an authorisation (AuthZ) request to determine whether the requested credential must be generated or not (fig. step 1).
2. The XACML PDP evaluates the AuthZ request using the defined XACML policies and sends back its verdict to the Capability Manager (fig. steps 2, 3).

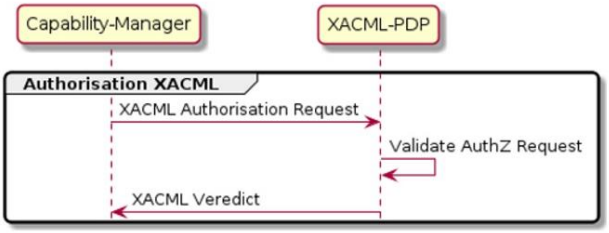


Figure 11: XACML PDP authorization flow diagram

7.2.2.3 Capability Manager

The Capability Manager is the component for generating capability tokens for the user in case of receiving affirmative authorization decisions from the XACML PDP of a request about an action or about the access to a resource. The Capability Manager signs the generated capability token that includes the client's public key and time restrictions associated with the specific policy delimiting the validity period for this credential.

The figure above shows a flow chart with a Capability Manager in an authorization process example:

1. When an authenticated user wants to get access to a resource or to perform an action an authorization request is sent to the Capability Manager (fig. step 1).
2. When this request, that includes the user's authentication (AuthN) token, is received by the Capability Manager it validates the token on the IdM getting back the user's identity attributes (fig. steps 2, 3) and then validates them (fig. step 4).
3. Once validated and with these attributes, the Capability Manager asks the XACML PDP sending an authorisation (AuthZ) request to determine whether the requested credential must be generated or not (fig. step 5).
4. The XACML PDP evaluates the AuthZ request using the defined policies and sends back its verdict to the Capability Manager (fig. steps 6, 7).
5. The Capability Manager generates then the Cap.Token and send it back to the user in a response (fig. steps 8, 9).

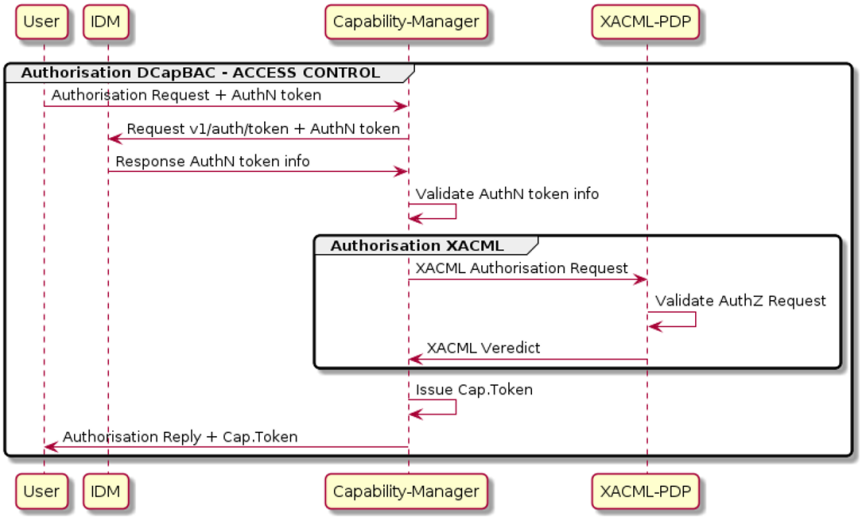


Figure 12: Capability Manager with capability token flow diagram

The format of the capability token is based on JSON as it can provide a simple, lightweight, efficient, and expressive data representation, which is suitable to be used on constrained networks and devices in IoT scenarios. The next text shows an example of capability token in JSON format:

```
{
 "id": "nlqfnfa6nqrlbh9h7tigg28ga1",
```

"ii": 1586166961,

"is": "capabilitymanager@odins.es",

"su": "Lucas",

"de": "https://153.55.55.120:2354",

"si":"MEUCIEEGwTGdlEeUxZv7jsh0UdWoLud3uqpMDvlT+GD7AiEAmwEuFHuG+XyRi9BEAMVPBIqRvOJlSIBkBT3K7LHCw=",

"ar": [

{

"ac": "GET",

"re":"/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:temperature.21"

}

],

"nb": 1586167961,

"na": 1586178916

}

- The identifier (ID): It is used to un-equivocally identify a capability token.
- The issuer (IS): Entity issuing and signing the capability token.
- The signature (SI): It carries the digital signature of the token.
- Access Rights (AR): The set of rights granted to the subject.
    - Action (AC): Its purpose is to identify a specific granted action ("get").
    - Resource (RE): The resource ("temperature") for which the action is granted.

7.2.2.4 PEP Proxy

The PEP (*Policy Enforcement Point*) is responsible for validating a generated assertion in an authentication token (X-AUTH-TOKEN) with the capability token that was already generated in a response by the Capability Manager to a user's authorization request. The PEP Proxy verifies that the public key contained in the received capability token is the same key that was used in the authentication process and verifies the token's signature by making use of the Capability Manager's public key. This component simplifies the access control mechanism to the resources, and it is a relevant feature on IoT scenarios since complex access control policies are not required to be deployed on end devices.

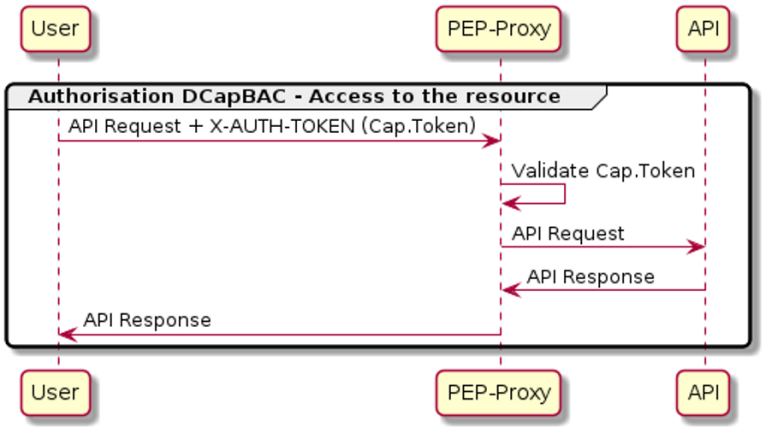


Figure 13: PEP Proxy flow diagram

In the figure above, a user that has already received the capability token from the Capability Manager attempts to access a resource. For this purpose, the user generates a request in which the Cap. Token is attached and that is handled by the PEP Proxy (fig. step 1) which validates the token (fig. step 2). After a positive validation, the PEP Proxy forwards the query to the system (Context Broker) (fig. step 3) as well as it forwards the response (fig. step 4) to the user (fig. step 5).

7.2.2.5 Traceability Agent (VICOM)

The authentication and authorization traceability component will log the access to DEMETER resources by logging the issue and use of authentication and authorization tokens. These tokens contain the information about the user who is logged to the system and the resources the user is intended to access.

The traceability agent will expose a REST API to register authentication and authorisation events (POST) and retrieve their details (GET). The REST API has been designed flexible enough to be able to use different traceability blockchain repositories (i.e. Quorum, HyperLedger Fabric, etc.)

The events logged will contain information about the receiver of the token, the sender of the token, the timestamp, the token details, and an optional data field to extend the information of the event.

The UML sequence diagrams are as follows:

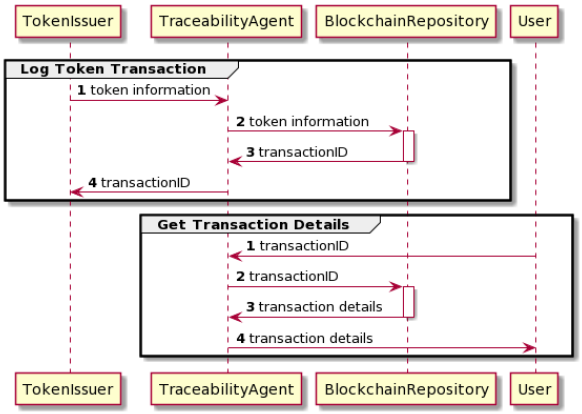


Figure 14: Traceability sequence diagrams

7.2.2.6 Traceability Blockchain Repository (VICOM)

A permissioned version of a blockchain *repository* has been chosen to provide the characteristics of immutability, privacy and compatibility required by the DEMETER Traceability Component. It supports both public and private transactions and smart contracts, and their states derived from a single, common, complete blockchain for transactions validated by every node in the network.

7.2.3 **Process View**

A user trying to access a DEMETER resource should first get authenticated at the Identity Manager to obtain and authentication token. Once the user is authenticated, the authentication token will be used to request access to DEMETER resources through the authorisation component, as described in the following sequence diagram:

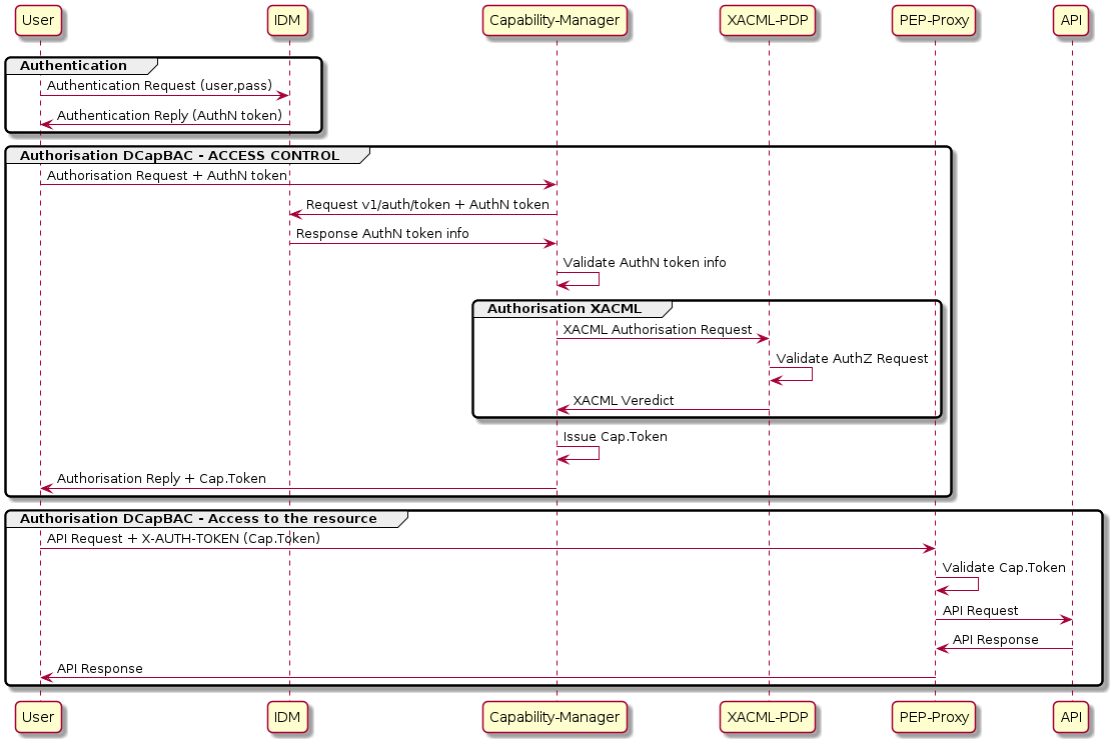


Figure 15: Authentication and Authorisation sequence diagram to access DEMETER resources

7.3 ***Identity Manager***

7.3.1 **Interfaces**

7.3.1.1 Data Models used in interfaces

The following data models are used by Keyrock to store the information for the application, user, organization, roles, and authentication tokens:

Table 2: User Data Model

Name	Keyrock User Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Username	String	sequence of characters that identifies a user
Description	String	text that provides further details about the user

Website	String	URL
Image	String	image to be used by an application representing the user
Gravatar	Integer	Gravatar image
Email	String (unique)	email provided by the user at registration
Password	String	string of characters, used to confirm the identity of the user
Date_Password	DateTime	date when the password was set
Admin	Integer	Boolean value indicating whether the user has administration rights
Extra	JSON	field where a JSON object can be stored to provided extra information

Table 3: Application Data Model

Name	Keyrock Application Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Name	String	string of characters that identifies the application
Description	String	text that provides further details about the application
URL	String	application's URL
Redirect_URL	String	URL required by the OAuth protocol
Redirec_sign_out_URL	String	the URL to which Keyrock will redirect a user if a sign out is performed from a service
Grant_Type	String	list of grant type authentication allowed for the application
Provider	String	Specify the provider of the application
Extra	JSON	field where a JSON object can be stored to provided extra information

Table 4: Organization Data Model

Name	Keyrock Organization Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Name	String	sequence of characters that identifies the organization
Description	String	text that provides further details about the organization
Website	String	URL provided

Table 5: Role Data Model

Name	Keyrock Role Data Model	
Property	Type	Description
Id	UUID	universally unique identifier
Name	String	sequence of characters that identifies the role

Table 6: Authentication Token Data Model

Name	*Keyrock* Authentication Token *Data Model*

Property	Type	Description
Access_Token	String (unique)	string issued by Keyrock as a token identifier
Method	String	specifies the grant type method used for the authentication
Expire_at	DateTime	Date and Time for the expiration of the authentication token

7.3.1.2 Description of APIs

In the following tables it will be provided the REST API details for the user to obtain a token from Keyrock using username and password, how to refresh that token and how to delete.

More information about Keyrock API can be found at:

- https://keyrock.docs.apiary.io/#introduction/preface/status
- https://swagger.lab.fiware.org/?url=https://raw.githubusercontent.com/FIWARE/specifications/master/OpenAPI/security.Idm/Idm-openapi.json

(REST API)

Title	Create token with Password
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://keyrock/v1/auth/tokens	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Data Params This field holds the body payload of a request.	
Required:	
"name"=[string]	Username set by the user (email)
Required:	
"password"=[string]	Password set by the user
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: Header: Content-Type:application/json,application/json; charset=utf-8 X-Subject-Token: 04c5b070-4292-4b3f-911b-36a103f3ac3f Content-Length:74 ETag:W/"4a-jYFzvNRMQcIZ2P+p5EfmbN+VHcw" Date:Mon, 19 Mar 2018 15:05:35 GMT Connection:keep-alive	Authentication token provided in the header (X-Subject-Token) and token details provided in the Body (method used to obtain the token and token expiration time)

Body:
{
"token": {
"methods": ["password"],
"expires_at": "2018-03-20T15:05:35.697Z"
}
}	

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400 Bad Request "Invalid grant: user credentials are invalid" "Invalid client: client is invalid"	 Error response for wrong username and/or wrong password. Error response for wrong client
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
curl --include \ --request POST \ --header "Content-Type: application/json" \ --data-binary "{ \"name\": \"alice@test.com\", \"password\": \"passw0rd\" }" \	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Refresh token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://keyrock/v1/auth/tokens	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Data Params This field holds the body payload of a request.	
Required:	
"token"=[string]	Token previously obtained
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: Header: Content-Type:application/json,application/json; charset=utf-8	Authentication token provided in the header (X-Subject-Token) and token details provided in the Body (method used to obtain the token and token expiration time)

X-Subject-Token: 65c6b870-3535-6b4f-345b-34a345f3ac7f
Content-Length:74
ETag:W/"4a-jYFzvNRMQcIZ2P+p5EfmbN+VHcw"
Date:Mon, 19 Mar 2018 16:05:35 GMT
Connection:keep-alive
 	
Body:
{
"token": {
"methods": ["password"],
"expires_at": "2018-03-20T16:05:35.697Z"
}
}	

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400 Bad Request "Invalid grant: refresh token is no longer valid"	The token provided is no longer valid, therefore, a new authentication token is not provided.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	

`curl --include \`
`--request POST \`
`--header "Content-Type: application/json" \`
`--data-binary "{`
`\"token\": \"token_id\"`
`}" \`	

Notes This field holds any additional helpful info related to this endpoint.	

Title	Revoke token
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://keyrock/v1/auth/tokens	
Method This field holds the type of the Method used	
DELETE	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Required:	
"X-Auth-token: auth_token"	Authentication token previously obtained for the user
Required:	
"X-Subject-token: subj_token"	Authentication token previously obtained for the user
Data Params This field holds the body payload of a request.	
Required:	
"token"=[string]	Token previously obtained

Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
204	Success response for token deletion
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400 Bad Request "Invalid grant: refresh token is no longer valid"	The token provided is no longer valid.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	

`curl --include \`
`  --request DELETE \`
`  --header "X-Auth-token: 65c6b870-3535-6b4f-345b-34a345f3ac7f" \`
`  --header "X-Subject-token: 65c6b870-3535-6b4f-345b-34a345f3ac7f" \`	

Notes This field holds any additional helpful info related to this endpoint.	

7.3.2 Technologies and implementation details

The Demeter Identity Manager has been implemented using the FIWARE Keyrock GE and it will be deployed (along with its database) using Docker containers.

7.4 *XACML PDP*

The XACML PDP manages the access control policies.

7.4.1 Interfaces

The PDP offers a RES interface to offer to the Capability Manager the verification of an authorization policy returning a verdict.

7.4.1.1 Data Models used in interfaces

In an XML format there is a PolicySet with a set of policies each one with an ID and a set of Rules, Subjects and Actions. The next table of properties shows the most relevant elements that are used by the PDP:

Name	XACML_Policy_Set	
Property	Type	Description
PolicySet.PolicySetId	String	PolicySet ID
Policy.PolicyId	String	Policy ID
Rule.RuleId	String	Rule ID
Rule.Effect	String	Rule effect required (permit/deny/…)
Subject.SubjectMatch.MatchId	String	Subject match XACML function. Examples: string-equal, etc.
Subject.SubjectMatch.AttributeValue.DataType	<any>	Subject value type as XMLSchema type. Example: string, number, etc.
Subject.SubjectMatch.AttributeValue.value	<any>	Subject value
Resource.ResourceMatch.MatchId	String	Resource match XACML function. Examples: string-equal, etc.
Resource.ResourceMatch.AttributeValue.DataType	String	Resource value type as XMLSchema type. Example: string, number, etc.

Resource.ResourceMatch.AttributeValue.value	String	Resource value as an entry point
Action.ActionMatch.MatchId	String	Action match XACML function. Examples: string-equal, etc.
Action.ActionMatch.AttributeValue.DataType	String	Action value type as XMLSchema type. Example: string, number, etc.
Action.ActionMatch.AttributeValue.value	<any>	Action value. Examples: “GET”, “PUT”, etc.

Next there is an example of an XACML policySet in XML format:

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:first-applicable"
PolicySetId="POLICY_SET">

  <Policy PolicyId="test1" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
algorithm:first-applicable">

    <Rule Effect="Permit" RuleId="001">

      <Target>

        <Subjects>

          <Subject>

            <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">

              <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Peter</AttributeValue>

<SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role"

 DataType="http://www.w3.org/2001/XMLSchema#string" />

            </SubjectMatch>

          </Subject>

        </Subjects>

        <Resources>

          <Resource>

            <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">

              <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">https://215.64.19.203:1020/ngsi-
ld/v1/entities?type=http://www.w3.org/ns/sosa/Sensor;idPattern=urn:ngsi-ld:Sensor:temperature.*

      </AttributeValue>

              <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
```

```
        DataType="http://www.w3.org/2001/XMLSchema#string" />
            </ResourceMatch>
          </Resource>
        </Resources>
        <Actions>
          <Action>
            <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">GET</AttributeValue>
              <ActionAttributeDesignator  AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-
id"
        DataType="http://www.w3.org/2001/XMLSchema#string" />
            </ActionMatch>
          </Action>
        </Actions>
      </Target>
    </Rule>
  </Policy>
</PolicySet>
```

7.4.1.2 ***Description of APIs***

Title	Obtain XACML PDP decision
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
/XACMLServletPDP/	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Data Params This field holds the body payload of a post request.	
Required:	
subject= [alphanumeric] <Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">	Subject of the resource's request. In DCapBAC scenario, it could correspond with a username (IDM). For example: "Peter"

<Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role" DataType="http://www.w3.org/2001/XMLSchema#string"> <AttributeValue>**subject**</AttributeValue> </Attribute> </Subject>	
resource= [alphanumeric] <Resource> <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" DataType="http://www.w3.org/2001/XMLSchema#string"> <AttributeValue>resource</AttributeValue> </Attribute> </Resource>	Resource: endpoint+path of the resource's request (protocol+IP+PORT+path). For example: "https://153.55.55.120:2354/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201". In DCapBAC scenario, endpoint corresponds with the PEP-Proxy one.
action=[alphanumeric] <Action> <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string"> <AttributeValue>action</AttributeValue> </Attribute> </Action>	Action: method of the resource's request For example: "POST", "GET", "PATCH", etc.
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
200 XACML – Permit	<Response> <Result ResourceID="resource"> <Decision>**Permit**</Decision> <Status> <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/> </Status> <Obligations> <Obligation ObligationId="liveTime" FulfillOn="Permit"> </Obligation> </Obligations> </Result> </Response>

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
200 XACML – NotApplicable	<Response> <Result ResourceID="resource"> <Decision>**NotApplicable**</Decision> <Status> <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/> </Status> </Result> </Response>
200 XACML – Deny	<Response> <Result ResourceID="resource"> <Decision>**Deny**</Decision> <Status> <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/> </Status> </Result> </Response>
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	

```
curl --location --request POST 'http://<PDP-IP>:8080/XACMLServletPDP/' \
--header 'Content-Type: text/plain' \
--data-raw '<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os">
  <Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
    <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>Peter</AttributeValue>
    </Attribute>
  </Subject>

  <Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>https://153.55.55.120:2354/ngsi-ld/v1/entities/urn:ngsi-
ld:Sensor:humidity.201</AttributeValue>
    </Attribute>
  </Resource>

  <Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>GET</AttributeValue>
    </Attribute>
```

</Action> <Environment/> </Request>'
Notes This field holds any additional helpful info related to this endpoint.

7.4.2 Technologies and implementation details

It is developed using XML and Java Servlets deployed on a Tomcat server.

7.5 *Capability Manager*

The Capability Manager is the endpoint for authorization requests.

7.5.1 Interfaces

The Capability Manager offers an interface to respond to an authorization request to access a resource or to perform an action. If the access is granted the capability token is sent back in the response.

7.5.1.1 Data Models used in interfaces

As the Capability Manager acts as an intermediate making a translation of an authorization request to an XACML authorization passing it to the XACML PDP, it has not an specific data model although, in a later stage after the PDP verdict, the Capability Manager creates a capability token which is a JSON signed document with the fields shown in the next table:

Name	Capability Manager Data Models	
Property	Type	Description
"id"	String	Identifier (ID). This field is used to un-equivocally identify a capability token.
"ii"	Numeric	Issued-time (II). It identifies the time at which the token was issued as the number of seconds from 1970-01-01T0:0:0Z.
"is"	String	Issuer (IS). Entity issuing and signing the capability token.
"su"	String	Subject (SU). The subject to which the rights from the token are granted. A public key has been used to validate the legitimacy of the subject.
"de"	String	Device (DE). It is a URI used to unequivocally identify the device to which the token applies.
“si”	String	Signature (SI). It carries the digital signature of the token.
"ar"	String	Access Rights (AR). This field represents the set of rights that the issuer has granted to the subject.
"ac"	String	Action (AC). Its purpose is to identify a specific granted action. Its value could be any CoAP method (GET, POST, PUT and DELETE).
"re"	String	Resource (RE).

		It represents the resource in the device for which the action is granted.
"nb"	Number	Not Before (NB). It expresses a time value. Before NB the token must not be accepted. Its value cannot be earlier than the II field and it implies the current time must be after or equal than NB.
"na"	Number	Not After (NA). It represents the time after which the token must not be accepted.

An example in JSON format is shown next:

{

"id": "nlqfnfa6nqrlbh9h7tigg28ga1",

"ii": 1586166961,

"is": "capabilitymanager@odins.es",

"su": "Peter",

"de": "https://153.55.55.120:2354",

"si":"MEUCIEEGwsTKGdlEeUxZv7jsh0UdWoFLud3uqpMDvnlT+GD7AiEAmwEu0FHuG+XyRi9BEAMaVPBIqRvOJlSIBkBT3K7LHCw=",

"ar": [

{

"ac": "GET",

"re":"/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"

}

],

"nb": 1586167961,

"na": 1586177961

}

*7.5.1.2 **Description of APIs***

Title	Obtain Capability Token.
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
/	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	

Required:	
Data Params This field holds the body payload of a post request.	
Required:	
token=[alphanumeric]	Subject of the resource's request. In DCapBAC scenario, it could correspond with an authentication token (IDM-KeyRock). For example: "753f103c-d8e5-4f4e-8720-13d5e2f55043"
de=[alphanumeric]	Endpoint: resource's request endpoint. Example, for a device (protocol+IP+PORT): "https://153.55.55.120:2354".
ac=[alphanumeric]	Action: method of the resource's request. Example: "POST", "GET", etc.
re=[alphanumeric]	Resource: path of the resource request. Example: "/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	

200
Body:
{
"id": "nlqfnfa6nqrlbh9h7tigg28ga1",
"ii": 1586166961,
"is": "capabilitymanager@odins.es",
"su": "Peter",
"de": "https://153.55.55.120:2354",
"si":"MEUCIEEGwsTKGdlEeUxZv7jsh0UdWoFLud3uqpMDvnlT+GD7AiEAmwEu0FHuG+XyRi9BEAMaVPBIqRvOJlSIBkBT3K7LHCw=",
"ar": [
{
"ac": "GET",
"re":"/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201"
}
],
"nb": 1586167961,
"na": 1586177961
}
	Capability Token
	"id": Identifier (ID). This field is used to un-equivocally identify a capability token.
	"ii": Issued-time (II). It identifies the time at which the token was issued as the number of seconds from 1970-01-01T0:0:0Z.
	"is": Issuer (IS). Entity issuing and signing the capability token.
	"su": Subject (SU). The subject to which the rights from the token are granted. A public key has been used to validate the legitimacy of the subject.
	"de": Device (DE). It is a URI used to unequivocally identify the device to which the token applies.
	Signature (SI). It carries the digital signature of the token.
	"ar": Access Rights (AR). This field represents the set of rights that the issuer has granted to the subject.
	"ac": Action (AC). Its purpose is to identify a specific granted action. Its value could be any CoAP method (GET, POST, PUT and DELETE).
	"re":" Resource (RE). It represents the resource in the device for which the action is granted.
	"nb": Not Before (NB). It expresses a time value. Before NB the token must not be accepted. Its value cannot be earlier than the II field and it

	implies the current time must be after or equal than NB. "na": Not After (NA). It represents the time after which the token must not be accepted.
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
500	"Can't generate capability token"
An IdM error code in case of error validating the authentication token (AuthN) on the IdM.	Text error associated to the IdM error code.
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
curl --location --request POST 'https://<CAPMAN-IP>:<CAPMAN-PORT>' \ --header 'Content-Type: application/json' \ --data-raw '{ "token": "753f103c-d8e5-4f4e-8720-13d5e2f55043", "de": "https://153.55.55.120:2354", "ac": "GET", "re": "/ngsi-ld/v1/entities/urn:ngsi-ld:Sensor:humidity.201" }'	
Notes This field holds any additional helpful info related to this endpoint.	

7.5.2 **Technologies and implementation details**

There is a Java implementation using Eclipse that can be used as an imported JAR file or via an API implemented in Python using Visual Studio Code that uses this same JAR file.

7.6 ***PEP Proxy***

The PEP acts as an intermediate between a user, service, device, etc. and the Information Repository (Broker) and also as a component of validation of the capability token.

7.6.1 **Interfaces**

The PEP acts as an intermediate and nowadays its implementation is integrated with NGSI or NGSI-LD Brokers (i.e. Orion, Scorpio) and, as a component of validation of the capability token, it is mandatory in both cases to include in the headers the X-AUTH-TOKEN that will include the capability token.

7.6.1.1 Data Models used in interfaces

The data model will be the same that the one in the implementation of NGSI or NGSI-LD used in the integrated Broker[1]. Also, as a component of validation of the capability token we need to include the structure of this token that was defined in section 7.5.

[1] NGSI, NGSI-LD Data Models: https://www.fiware.org/developers/data-models/

7.6.1.2 Description of APIs

As mentioned before, this component acts as an intermediate and will use NGSI or NGSI-LD APIs depending on the implementation used in the integrated Broker. These APIs are offered by Brokers as Orion (NGSI[2]) and Scorpio (NGSI-LD[3]).

7.6.2 **Technologies and implementation details**

The implementation is made in Python using Visual Studio Code.

7.7 ***Traceability Agent***

The authentication and authorization traceability component will log the access to DEMETER resources by logging the issue and use of authentication and authorization tokens. These tokens contain the information about the user who is logged to the system and the resources the user is intended to access.

7.7.1 **Interfaces**

7.7.1.1 Data Models used in interfaces

Name	Traceability Agent Data Model	
Property	Type	Description
Receiver	String	user that obtain the right to access a Demeter resource
Sender	String	identification of who is issuing the right
Timestamp	DateTime	time of occurrence of the event
OptionalData	JSON	optional data to extend the information of the registered event

7.7.1.2 Description of APIs

Title	Register Event
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool /v1/send	
Method This field holds the type of the Method used	
POST	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Required:	
Sender=[string]	identification of who is issuing the right
Optional:	
Recipient=[string]	the beneficiary of the right
Optional:	

[2] Orion, NGSI API: https://fiware-orion.readthedocs.io/en/master/user/walkthrough_apiv2/index.html
[3] Scorpio, NGSI-LD API: https://docbox.etsi.org/ISG/CIM/Open/PDF-Copy-of-GS-CIM-009-NGSI-LD-API-V1.2.1-with-line-numbers.pdf

Payload=[JSON]	Token information payload with the details of the transaction
Optional:	
OptionalData=[JSON]	Optional data
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {“transactionId”=”123e4567-e89b-12d3-a456-426614174000”}	response description value of a key as a transaction ID
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400	Error response
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
curl --location --request POST ' http://audit_tool /v1/send ' \ --header 'Content-Type: application/json' \ --data-raw '{ "sender": “753f103c-d8e5-4f4e-8720-13d5e2f55043”, "recipient": “534f503d-f8e6-5g7e-1234-53d8g4d55043”, "payload": {“authentication_token”=“753f103c-d8e5-4f4e-8720-13d5e2f55043”, “timestamp”="2018-03-20T15:05:35.697Z"} "optionalData": {} }'	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Transaction Details
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://audit_tool/v1/transaction/{hash}	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
N/A	parameter description
Optional:	
N/A	parameter description
Data Params This field holds the body payload of a request.	
Required:	
transactionId=[string]	Key value as a transaction ID
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: '{	Transaction details

"sender": “753f103c-d8e5-4f4e-8720-13d5e2f55043”, "tecipient": “https://153.55.55.120:2354”, "payload": {“authentication_token”=“753f103c-d8e5-4f4e-8720-13d5e2f55043”, “timestamp”="2018-03-20T15:05:35.697Z"} "optionalData": {} }	
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
400	Error response, transaction ID does not exist
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
curl --location --request POST ' http://audit_tool /v1/send ' \ --header 'Content-Type: application/json' \ --data-raw '{ "transactionId": “123e4567-e89b-12d3-a456-426614174000”}'	
Notes This field holds any additional helpful info related to this endpoint.	

7.7.1.3 Technologies and implementation details

The traceability agent has been implemented using Django REST Framework and it will be deployed using Docker containers.

8 DEMETER Enabler Hub

8.1 *Description*

The DEMETER Enabler HUB (DEH) is one of the most crucial components of the DEMETER project; it represents the digital space dedicated to end users of DEMETER where they will be able to create and register their own resources. Users will have two roles and they will act as DEMETER Provider and DEMETER Consumer. A DEMETER Provider will be able to offer his resources (components, services, data sources, devices, platforms, etc), while DEMETER Consumers will be able to browse it, and find suitable resources matching their requirements. In order to support this, the DEH involves communication between various DEMETER components. Taking this into account, each component inside DEH will expose endpoints through their internal APIs, and all data will be based on the AIM model (JSON-LD interoperability exchange format). Data entities from any Platform, Thing, Application, Service will be managed through these APIs, but for the sole purpose of discovery and management of the resource registry maintained by the DEH. To make the solution more flexible and easier to maintain, all components inside the DEH will be developed as separate services and deployed as standalone Docker containers. The DEH Dashboard (DEH Dymer sub-component described below) will be provided as an external component outside of the Core Services which will be available as SaaS (Software as a Service) and consists of the Compatibility Checker, Resource Registry Management and Discovery Management as shown in the figure below:

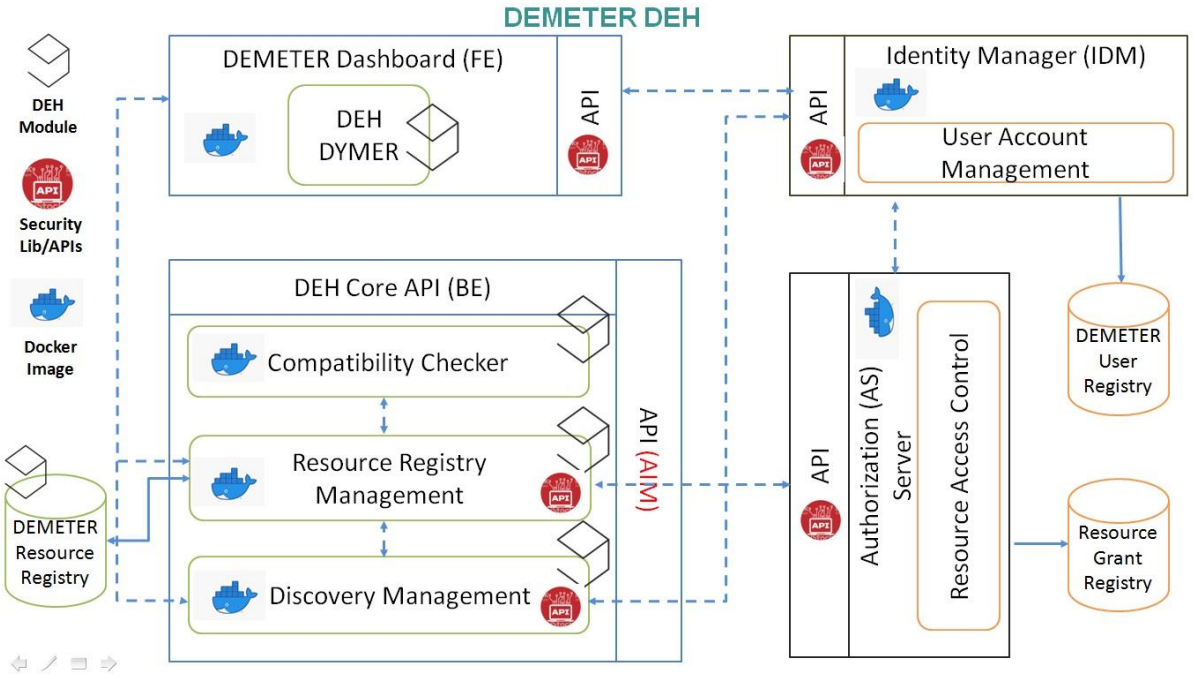


Figure 16: DEH Functional Architecture

Secured communication among all components will be provided by a Security Enabler, more specifically by the Identity Manager, and Resource Access Control components. The DEH Enabler will communicate with the Brokerage Service Environment (BSE) where all data gathered from DEMETER Pilots will be published. After the publishing, resource data (in AIM format) should be verified using the Compatibility Checker component and if data satisfies all necessary requirements, it will be passed to the Resource Registry Management component, which is in charge of all operations related to storing and managing DEMETER Entities. At the end, the DEH Dashboard will be able to show these resources in resource register mode to the end users of DEMETER, who intend to view them.

All Hub components will be made available for deployment via docker containerization. This will increase the configurability of the APIs and the flexibility of the DEH components by allowing different deployment modes such as cloud centric or Pilot environments.

8.2 *Development View*

The development view, also known as an implementation view, depicts the system from an engineer's point of view with a focus on the software module organization. For the purpose of representing the Development view, UML Component Diagram will be used to depict building blocks, components and their internal modules.

8.2.1 **Component diagram**

The component diagram, which is also known as a UML component diagram, is used to represent the physical aspect of a system and depicts the organization and the connection between internal components.

Figure 17 shows the component diagram that depicts the decomposition of the DEH into Building Blocks and the relations between them.

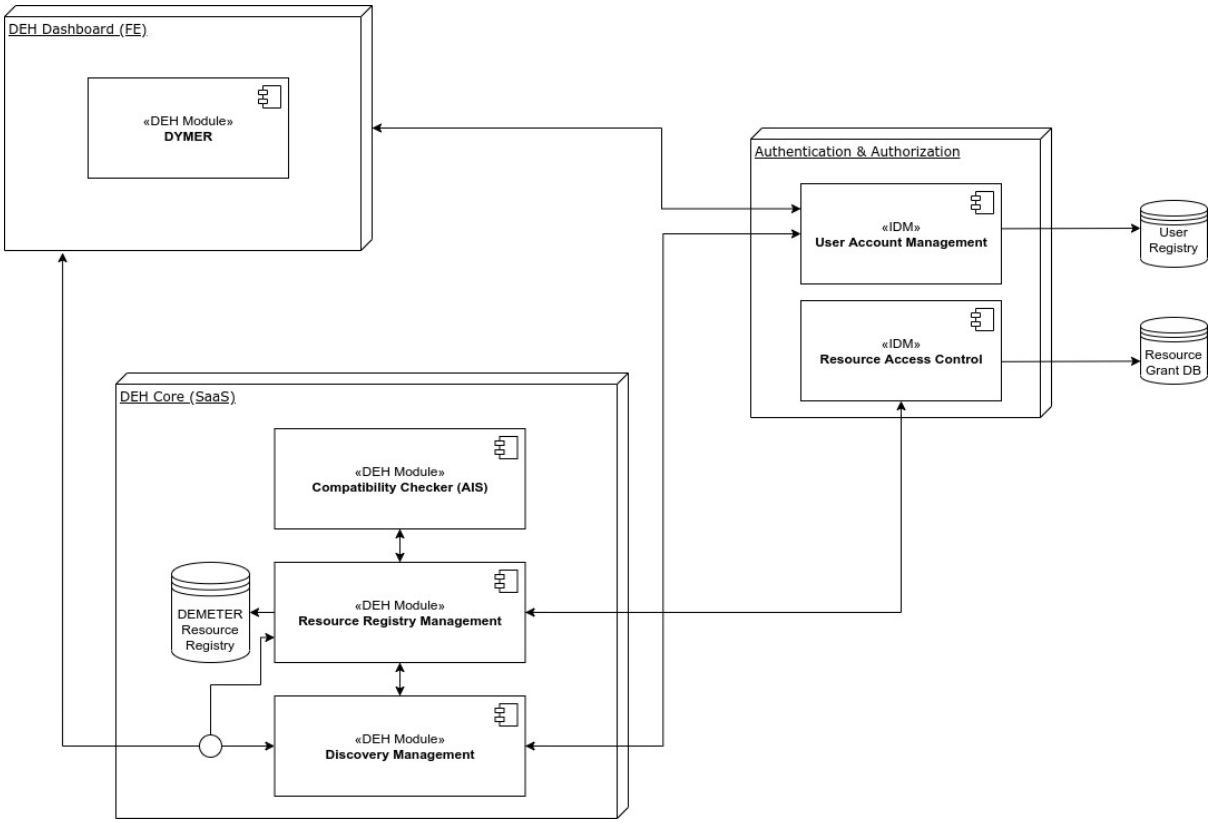


Figure 17: DEMETER Enabler HUB component diagram

DEH Components is composed of three main modules:

- **DEH Dashboard** functional module is in charge for User Interaction & Data Visualisation. It will allow users to login to DEH, discover, register and manage DEMETER Enablers.

- **DEH Authentication & Authorization** functional module is in charge for User authentication and authorization. It will contain information related to users, such as personal data, credentials, but also access policies.
- **DEH Core** functional module is in charge for managing DEMETER Resources. It will manage creating, editing, deleting and Discovery of DEMETER resources.

8.2.2 **Building blocks (components)**

This section contains detailed descriptions of components inside the DEH. Each component will be presented in subsections, which contain information about it, its purpose inside the DEH and component diagrams which depict their internal functional modules.

8.2.2.1 DEH Dashboard

DEH Dashboard represents the DEH front-end application, which will be used by end-users or DEMETER Stakeholders for resource creation or discovery. The DEH resources are represented by a set of entities such as Component, Device, Service, Dataset, Platform (and all those that will be added later in the project) which can be added via the DEH Dashboard or web-based UI (User Interface).

This fronted application for Stakeholders will be provided by ENG, which in the context of the DEMETER project for this specific component will use technology developed within its research laboratories, namely the DYMER. The DYMER is an open source suite for resource catalogue visualisation. DYMER provides advanced mapping capabilities between a data model in JSON format and its graphic template on the one hand, and on the other hand it provides a JavaScript framework for integrating the DYMER template into a web-based application. The software is flexible because it adopts open technologies and can be used in various environments without considerable requirements.

The DYMER consists of two main components:

- DYMER-Core (or DYMER business service module)
- DYMER-Viewer

DYMER Core is based on microservice architectural style with an approach to develop a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms using HTTP/REST protocols alongside JSON.

The diagram in Figure 18 depicts the DEMETER Enabler HUB Dashboard building block components:

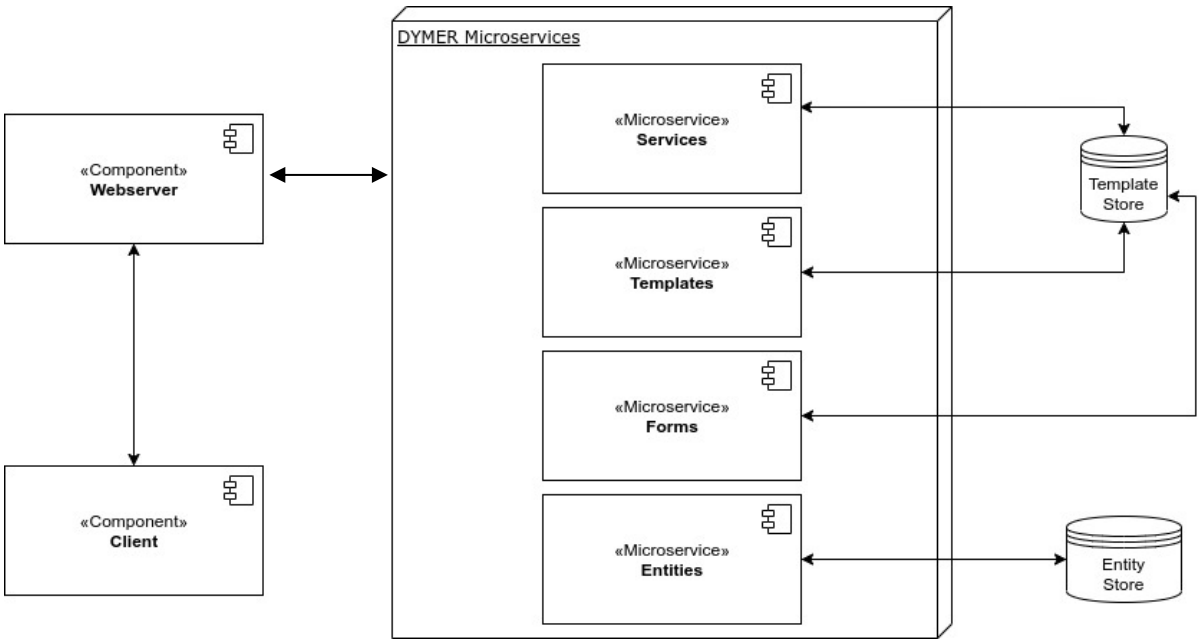


Figure 18: DEMETER Enabler Hub Dashboard building block components

Each microservice is developed with a specific role, however among the main ones we can identify three that have most impact on DEH:

- **Templates microservice** is responsible for generating graphic templates that can be used in order to display products and services using logic-less templates.
- **Forms microservice** is responsible for modelling data and metadata inherent to the products and services offered in DEH.
- **Entities microservice** is responsible for managing the storage and usage of the product and its services.

These microservice are developed with Express.js[4] framework for Node.js[5], designed for building web applications and APIs, released as free and open-source software under the MIT License[6].

The information is stored in NoSQL[7] Database that provides high performance, high availability, and automatic scaling. Service-Entities use Elasticsearch[8] that is a distributed, open source search and analytics engine for all types of data, including textual, numerical, geospatial, structured, and unstructured that stores data in JSON format.

Interaction with the **DYMER-Core** takes place through the **DYMER-Viewer** that is a fast, small, and feature-rich JavaScript library. Thanks to it, it is possible to interact with the platform facilitating the user in the use of data by offering a single search point and displaying the results in special graphic templates.

[4] https://expressjs.com/it/
[5] https://nodejs.org/it/
[6] https://opensource.org/licenses/MIT
[7] https://en.wikipedia.org/wiki/NoSQL
[8] https://www.elastic.co/

When it comes to the security part, DYMER will communicate with User Account management and Resource Access Control components. These interactions will be covered in section 8.3. Process View.

DYMER created the frontend technology that allows to create the Dashboard user for the DEMETER resource catalogue. The DEH end-user can register themselves to the DEH catalogue through a specific registration form made available by the Fiware-IdM component, which will be integrated with through its API. At the end of the registration the user will be able to access to the DEH resource catalogue using single sign-on authentication service as shown in the figure below:

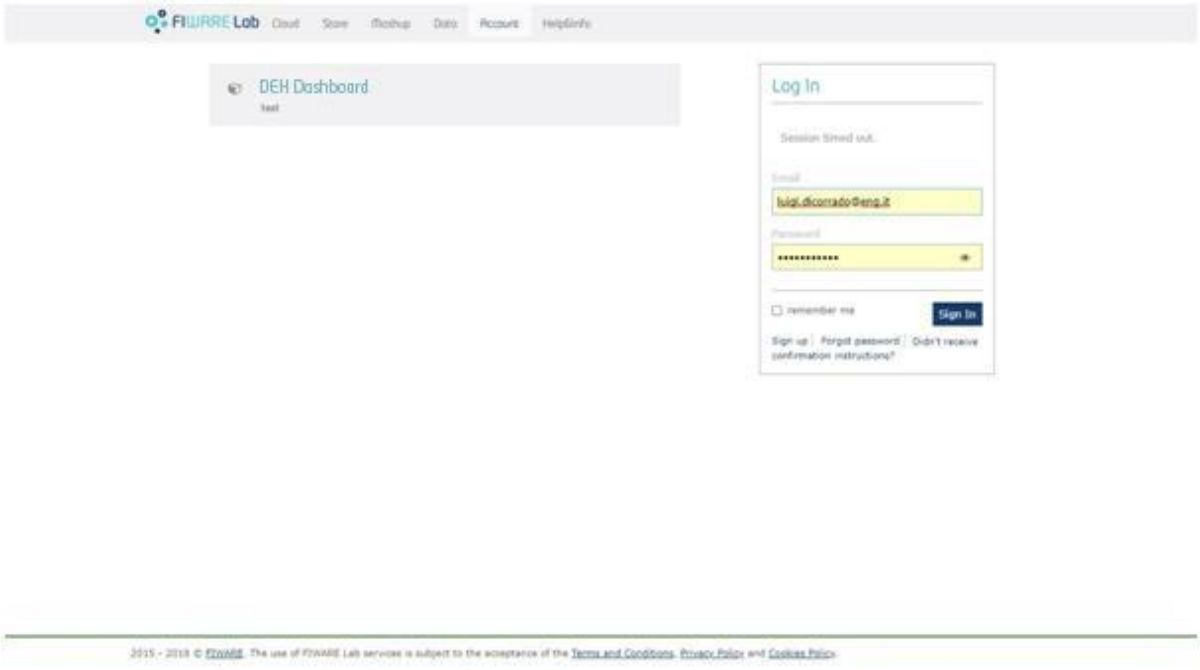


Figure 19: DEH Dashboard user login

The user entering the credentials (Username and Password) can access to the DEH catalog of resources and to a whole series of features that are in the design and implementation phase.

In order to have a shared idea about the DEH features, a survey was prepared by ENGINEERING with the support of Fraunhofer (WP7 “Multi-Actor Ecosystem Development” leader) and T3.5 participants, then reviewed by WP leaders and cluster pilot leaders.

The survey proposed a list of possible features for the DEH based on the Grant Agreement (GA) and aimed at finding a prioritization for the implementation of those features; Moreover it gave the possibility to add additional comments on proposed features and suggestions for any other missing features.

The survey was sent in December (through the online survey-management system, EU survey[9]) to WP3 participants, considering that DEH users will be mainly developers and 16 answers were collected.

The collected answers were used as “user requirements” for the design of the DEH and can be identified in the following:

[9] https://ec.europa.eu/eusurvey/home/welcome

- User profile management
- Resource discovery through search API
- New resources creation and editing
- Resource compatibility checking
- Resource rating visualisation

The DEH Dashboard is under construction, but below are described, through some screenshots, the features currently available in the HUB. Figure 20 show the dashboard that appears to the user when the latter accesses the resource catalogue (obviously the image shows only sample data). The user will be able to navigate the list of resources, which will be paged if the number of resources exceeds a specific threshold. In the list, the user displays a first detail of each resource (such as Component, Device, Service, Dataset, Platform) which shows only some data such as the name of the resource, the description and the endpoint to view or download the selected resource.

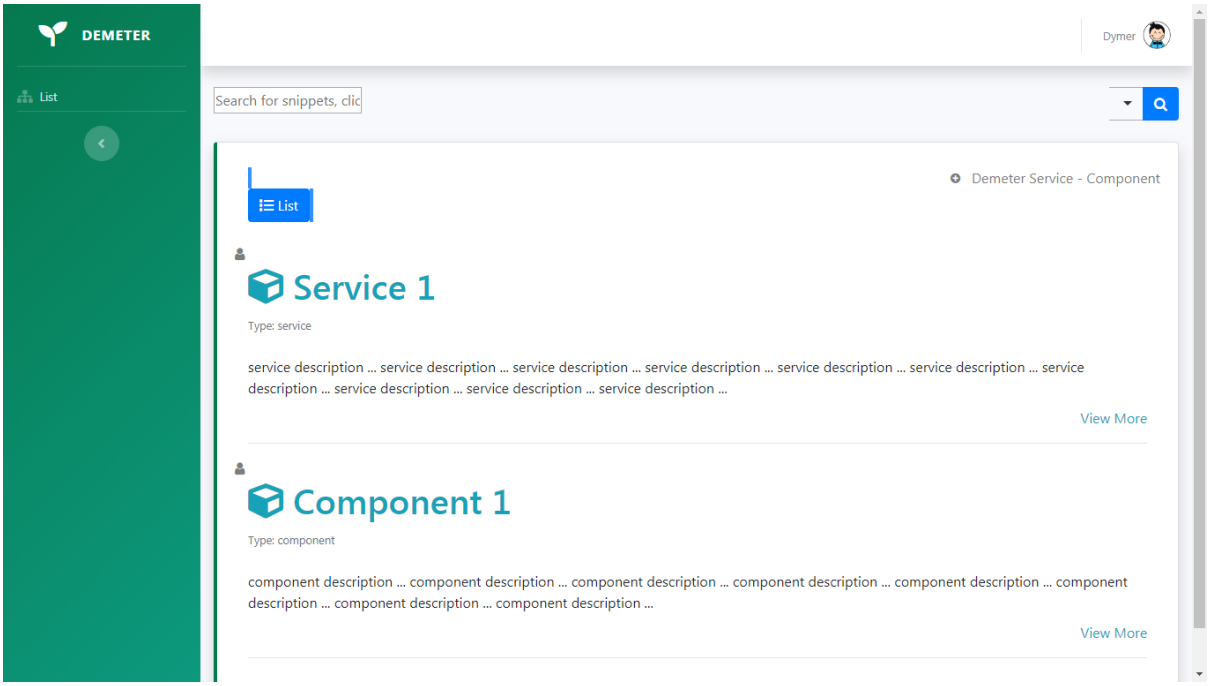


Figure 20: DEH Dashboard – Resource catalogue

The user will also be able to perform a search among the resources in the list, using a special search function. The search will obviously be gradually refined in the implementation, in its technical details and as a web module by adding the necessary filters that will become necessary from time to time. Figure 21 shows how a user can access the search filters made available by the dashboard to perform targeted searches on the resources contained in the catalogue:

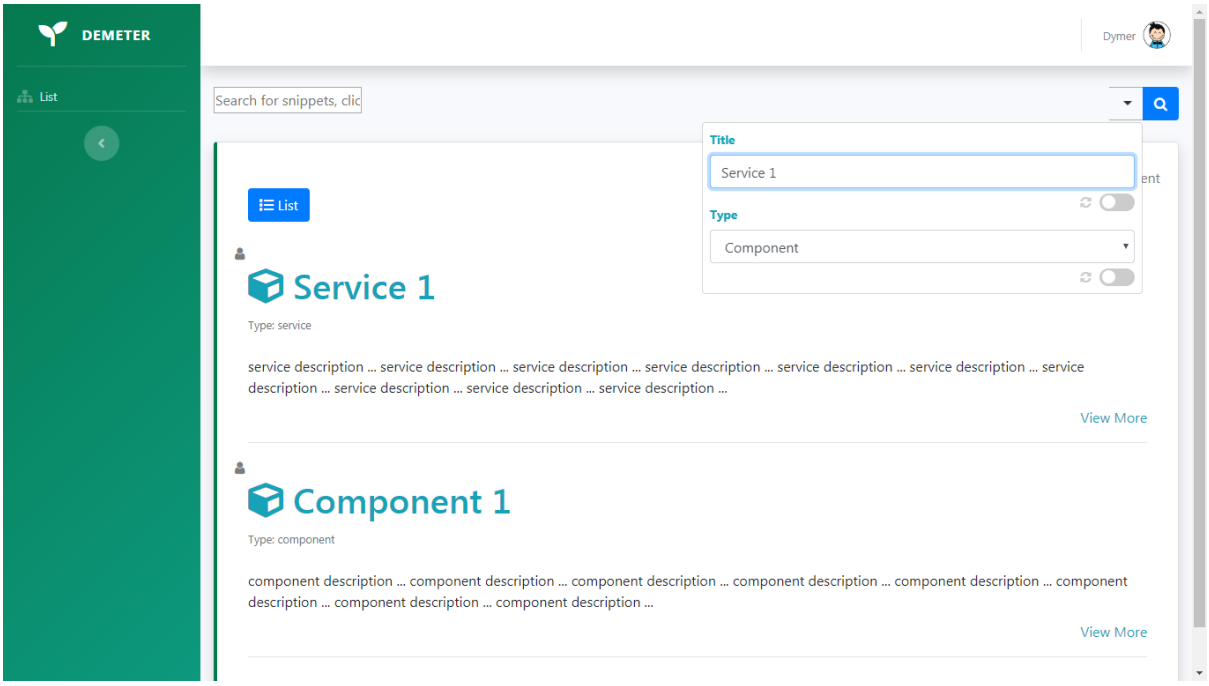
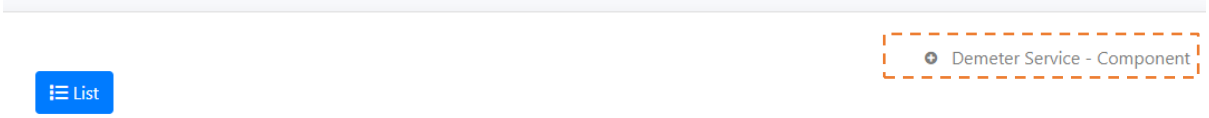


Figure 21: DEH Dashboard search filter on resource catalogue

A DEH user can insert a new resource and edit the one already existing. This is possible through “add” or “edit”/“upgrade” functions. To access the function of creating a new resource, it is needed to click on a specific image in the upper right part of the list:



By clicking on the "+" button the user can access a popup that shows the data entry form for the resource:

Figure 22: DEH Dashboard – create new entity/resource

The user can define a whole series of details relating to the resource he intends to create, including the name, the type, the description and a web link to the resource (since it concerns a web reference to a component for its download or a service endpoint), the title, the status and finally the visibility (this property is related to the need of the user to make the resource public or discoverable by the other users of the HUB or private). It is important to remark here that this is still an evolving job, so this form will be updated in the near future to match end users' needs. This work, which involves continuous updates of the resource model, will be well supported by DYMER, which uses dynamic module models associated with the resource data model. The more frequently the data model changes, the more the DYMER will be able to adapt the user interfaces to the new model.

Finally, the user can edit the data associated with the individual resources at any time using the “View More” link in the list. For each resource in the list the user can access to specific resource by clicking on related “View More” link. The screenshot below shows what the user sees after clicking on the link:

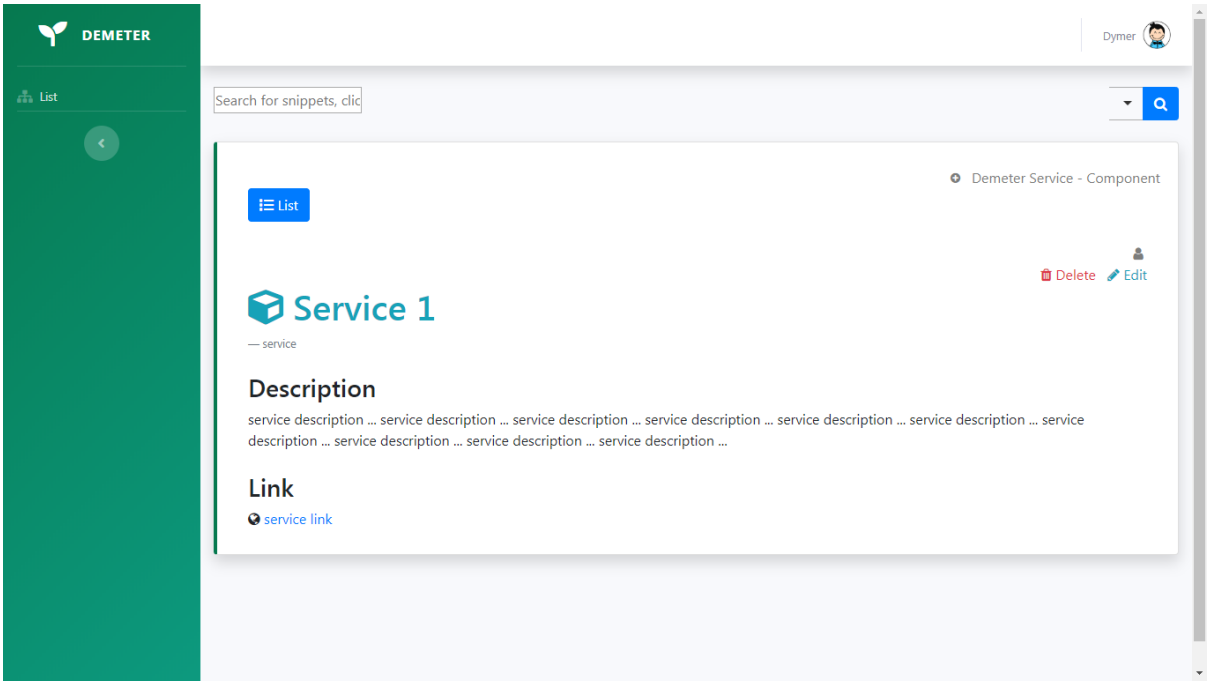


Figure 23: DEH Dashboard – List of entities/resources

The user can decide to delete the resource from the register or simply to edit it (only if the user is the owner of the selected resource). In the first case, the user must click on the "Delete" link and wait for the system to send a visual message via pop-up with the result of deleting the selected resource from the resource registry.

Figure 24 shows the form that the user accesses from the list of resources by simply clicking on the link “Edit”:

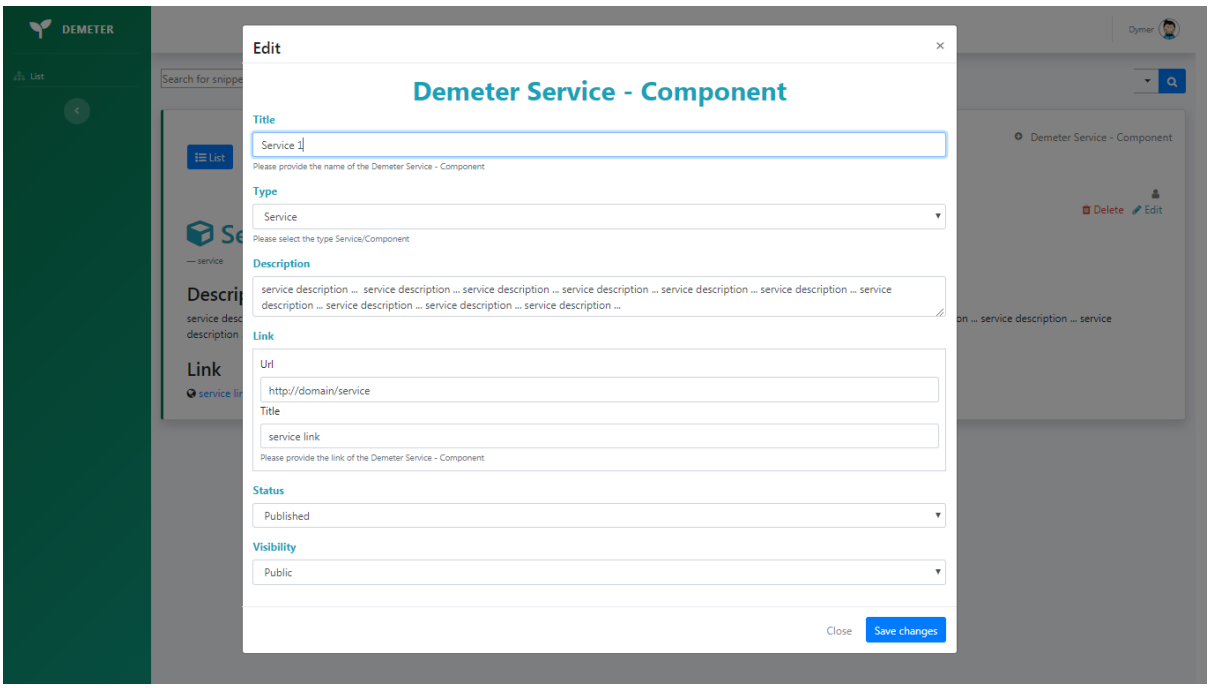


Figure 24: DEH Dashboard – edit existing entity/resource

Changing a resource means changing all the data previously defined in the creation phase, allowing the user to correct any errors such as bad typing of information.

The DYMER also implements administration functionality represented by a web-based application, external to the DEH catalogue, to allow a user with Admin role to have complete management of Templates, Models or Forms and Entities. Figure 25 shows administration dashboard of DYMER component:

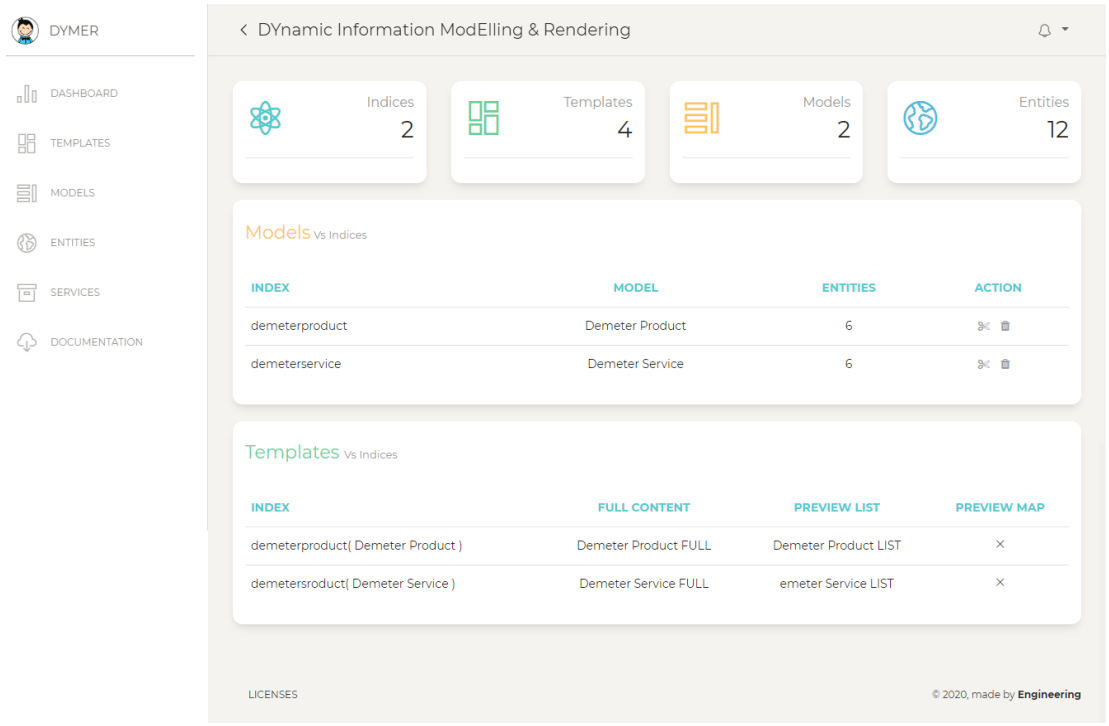


Figure 25: DYMER administration dashboard

By clicking on the Templates link menu, on the left in the drop-down list, the user can access to the list of the currently registered Templates, in order to view them or create new templates through “Manage Template ” functionality or modify the existing ones. Figure 26 shows the Template list:

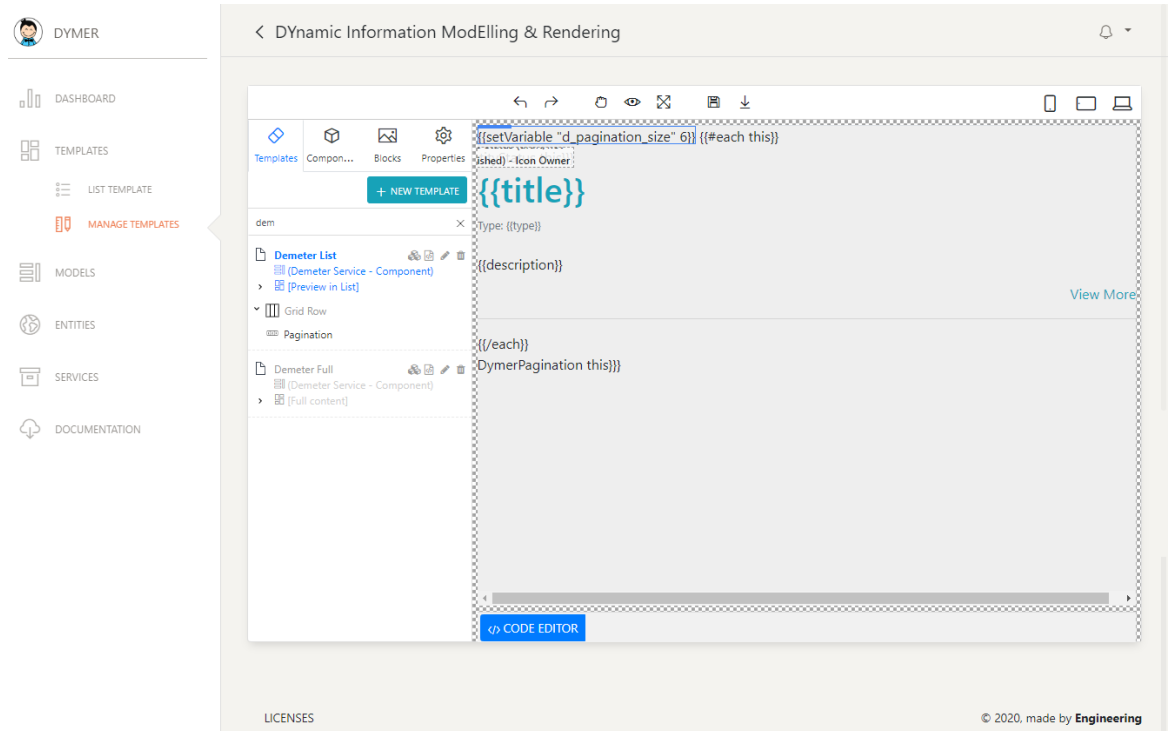


Figure 26: DYMER administration – template management

The same management features are available respectively for the models/forms and for the entities as the figures below shown:

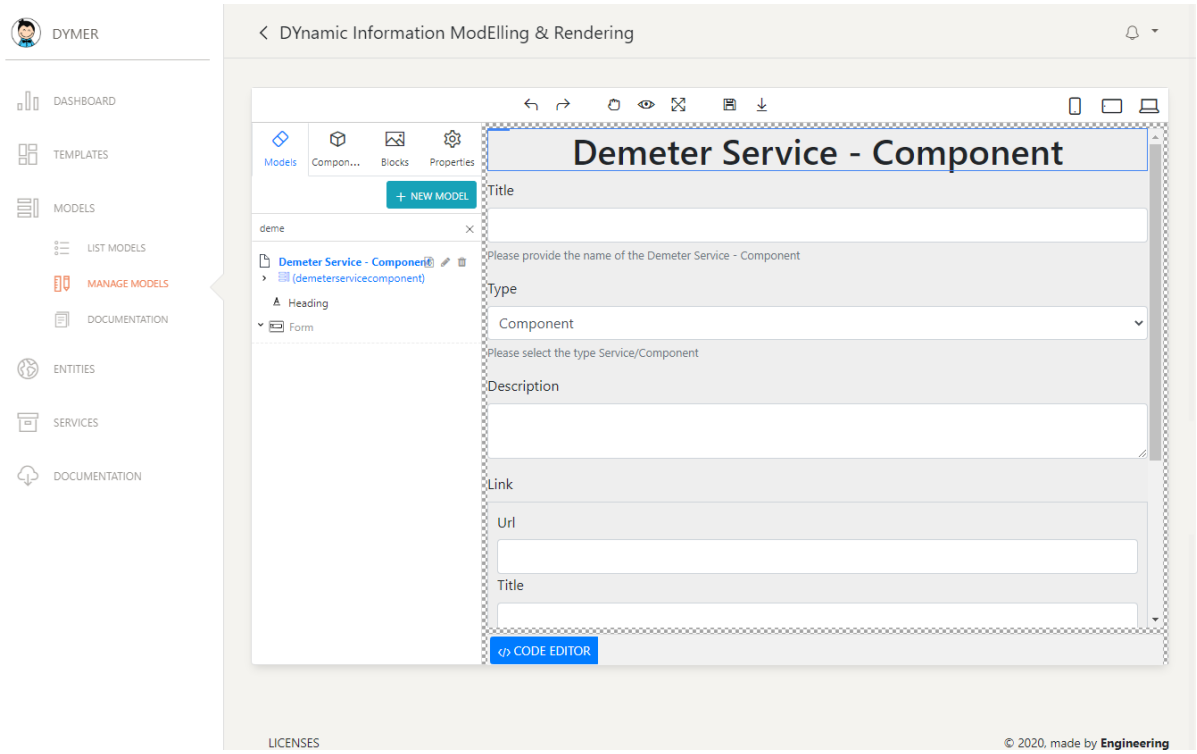


Figure 27: DYMER administration– models/forms management

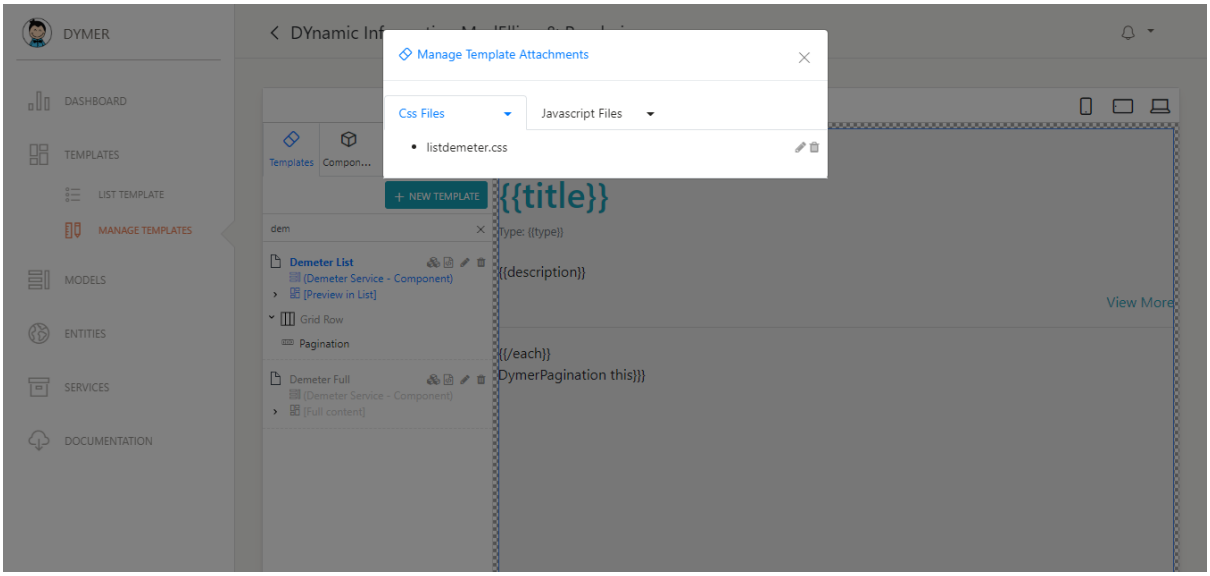


Figure 28: DYMER administration– models/template manage css/javascript

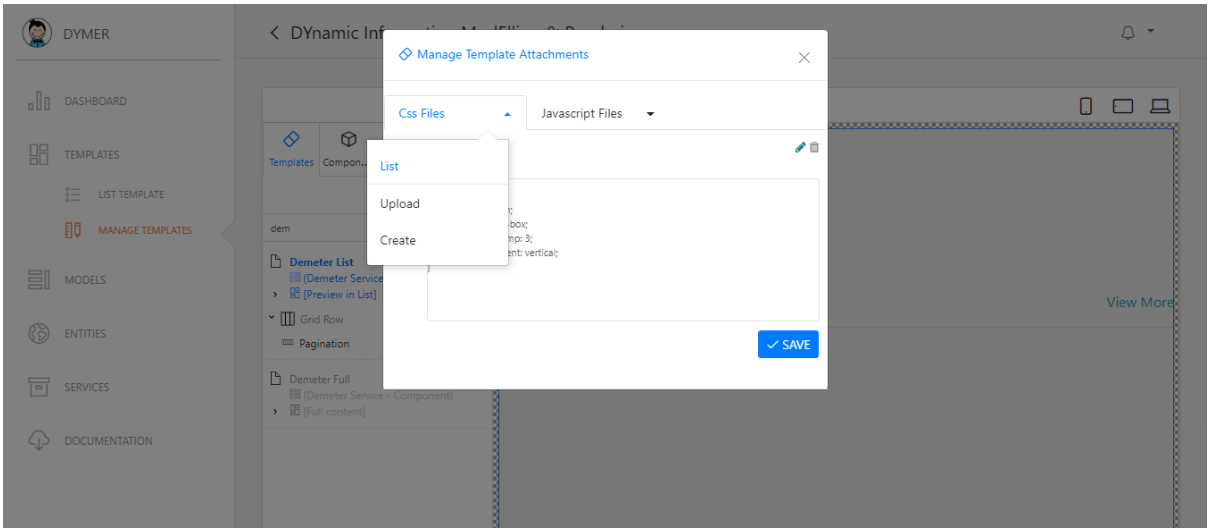


Figure 29: DYMER administration– models/template crud css/javascript

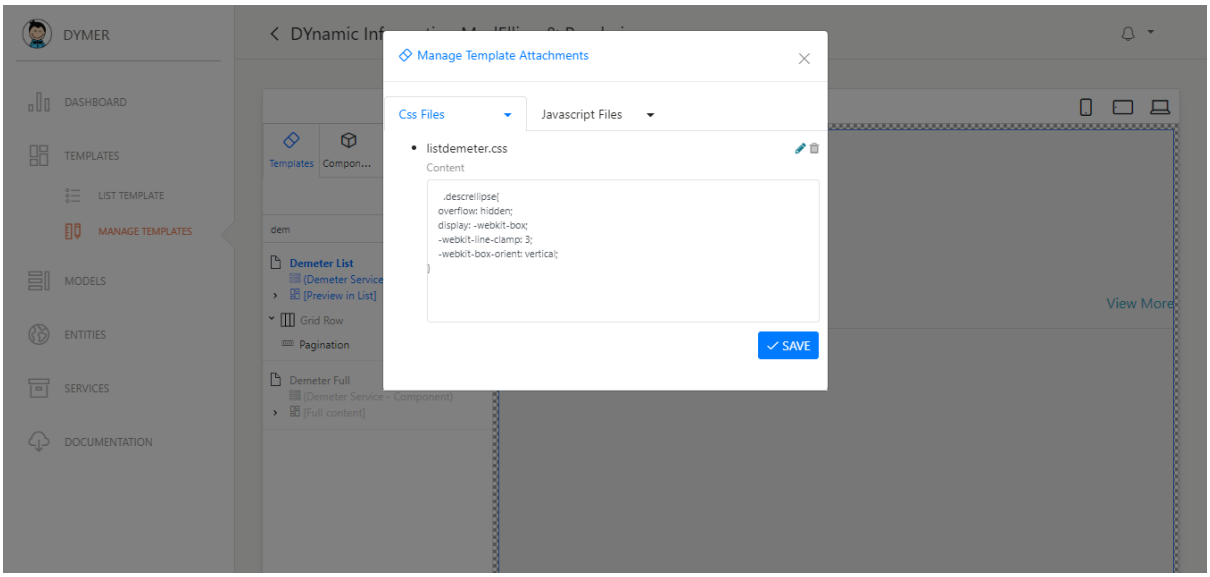


Figure 30: DYMER administration– models/template manage edit css/javascript

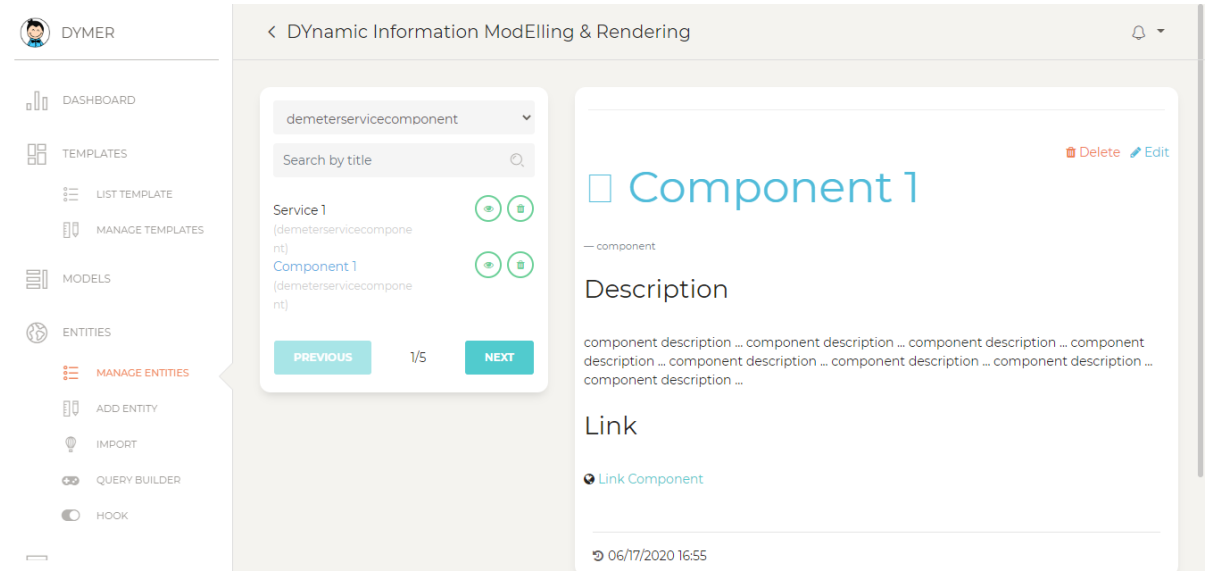


Figure 31: DYMER administration– entities management

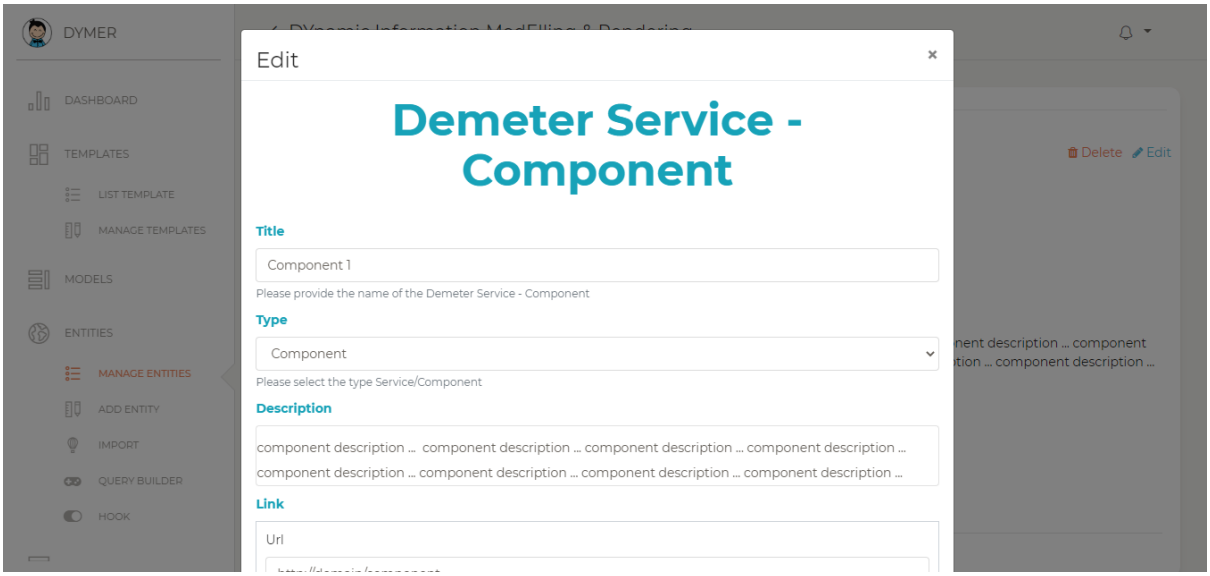


Figure 32: DYMER administration – edit entities

8.2.2.2 DEH Resource Registry Management

DEH Resource Registry management component represents essentially a master service included in the DEH Core services. It is the only component inside the DEH which has direct access to the DEH Resource Registry and thus the component will act like a Data Access Object (DAO) Layer. The functionalities that DEH Resource Registry Management module should provide consist in managing and monitoring resources. As will be described in the next subsection, this module also works closely and is used by the DEH Resource Discovery component.

Managing resources includes the functionalities which will provide storing new resources, as well as updating the properties of existing ones and deleting them when they are no longer offered. Beside the mandatory information stored about any resource offered by the DEH, this component will be

able to provide and store information about the changes that were made to the registry, including their time and version. Part of the process for registering (or updating) a resource also includes passing the relevant information (and potentially) part of the code or the interface data of the resource to the compatibility checker component (described in section 8.2.2.4) in order to verify compatibility with DEMETER and AIM; only after this check is completed successfully can any resource be registered.

In addition, this component will work closely with the Resource Discovery module (see section 8.2.2.3), as the later will query for specific resources registered in the Resource Registry DB. To facilitate these queries the audit component of this module will also query the resource access control policies and data as stored in that component of the DEH in order to determine which resources a given user is allowed to access. This information is provided by the user who registered the resource and is stored in the Resource Registry DB together with the remaining data regarding any resource.

Finally, the Resource Registry Management will also provide the information and enable the usage of the stored resources to the end users.

As part of the last functionality, the usage of the resources by end users, it will offer tools for monitoring these resources and this includes functionalities which will record the history regarding the consumption of specific enablers and resource and their work load.

Figure 33 shows a diagram that depicts internal functional modules of DEH Resource Registry Management and communication with other DEH modules.

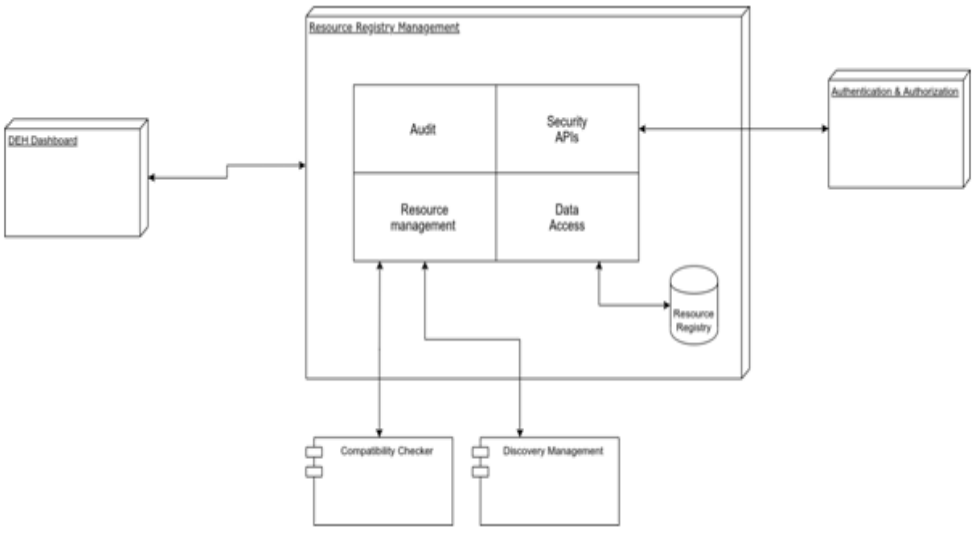


Figure 33: Resource Registry Management building block diagram

Resource Registry management is composed of four internal modules:

- Audit - responsible for collecting data related to resource, including their capabilities, their access information as well as user access control policies.
- Data Access - responsible for accessing the actual data for a resource as stored in the Resource Registry DB.
- Resource Management - responsible in general for the resource management process and for interfacing with other components such as the compatibility checker and the discovery management.
- Security APIs – responsible for providing authentication and authorization.

The Audit module provides support for collecting data about resources such as the date and time of their creation or any edit or update to the resource, its rating, its usage, and its history of consumption. In addition, it will be responsible for gathering the information needed by other modules such as access policies to the resource as well as the data needed to verify compatibility. The Audit module communicates all this data to the Resource Management, in order to pass all of that information along to other components.

The Data Access module provides support for direct access to the Resource Repository DB in order to manipulate the data stored about a registered resource allowing to store, read, edit and delete its relevant data.

The Resource Management module is the only module that has direct communication with external modules like the Compatibility checker in order to validate if a resource is compatible with DEMETER or the Discovery Module in order to find proper resources based on user queries (filters). This module also provides support related to preparing a resource for storing. After the resource is checked, and prepared, data is passed to the Data Access modules in order to store it inside the Resource Registry DB. Finally, the resource management module allows searching and accessing resources based on the following criteria: resource uid, type, tags, category, and text search. The research capabilities of this module may be integrated and improved from time to time during the development phase of the DEH, in order to support new application needs as much as possible or to cover new integration requirements with the other DEMETER software components.

For purposes of the **Resource Registry store**, a NoSQL document-oriented database is used. NoSQL systems are generally more efficient than relational databases because of their ease of management. Since the information to be collected, stored, consulted essentially refer to DEH resources metadata, the challenge will be to use a NoSQL technology for this purpose.

The DEH Resource Registry: represents the solution to the metadata store based on NoSQL data archives. DEH Resource Registry works as part of store data and take advantage of the underlying NoSQL functionality to provide its services. This database will face a number of important challenges:

- first, gathering metadata without imposing too much overhead for CRUD operations arriving from client application,
- second, it must allow a user to specify (via API) what the metadata collected and how to use them.
- Finally, it must manage the size of the stored data, the number of incoming CRUD operations and obviously the size of the metadata itself.

Probably if the size of the metadata and the operations managed by the NoSQL database (in the continuous collection of metadata), should cause a great overload of the system, configuration-level facilities could be introduced which would allow a more moderate acquisition of metadata by sizing the read and write operations performed by client applications on the NoSQL store.

The DEH Resource Registry is a table for registering objects for which the metadata will be collected. Each resource is identified by some attributes such as for example *uid, name and resource ownership*. The metadata will be collected and stored in this table (resource_metadata). Client applications will only be allowed read-only access to this table to protect the metadata scheme. To allow a client to modify a resource, all authorizations will be verified through DEMETER's security enablers.

The Security APIs module provides support for connection with Authentication Authorization block, in order to get respective access policies.

The security module in the authentication authorization block provides the solution for controlling the access to the stored resources.

The Security API implements OAuth2 for the implementation of the Identity Management KeyRock (authentication) and a REST API for the implementation of DCapBAC (authorization).

The DCapBAC REST API end point function returns the authorization or Capability Token. A definition of this is shown next, definition that can be found in more detail in this document Section 7.4:

Name: generateCapabilityToken (authtoken,subject,resource,action)
Expected output: Capability Token (a signed JSON document)
Error messages: "Error connecting to the Authorization server"
Data model used: each of the parameters received by this function are strings

The description of these parameters is:

Name	generateCapabilityToken()	
Property	**Type**	**Description**
authtoken	String	The authentication token obtained from the Identity Management
subject	String	The subject of the authorisation query
resource	String	The resource intended to be accessed
action	String	The operation mode: GET, POST, PUT, PATCH or DELETE

Sample call:

generateCapabilityToken("04c5b070-4292-4b3f-911b-",user@dem.com,"ngsi-ld/v1/entities","GET")

Once the DCapBAC REST API has been called and the authorization token returned, it will be included in the access to the resource in the corresponding repository. Before performing the authorization process, the authentication one must have been done as it generates an Authentication Token that must be included in the authorization calls to be validated.

The authorization process, in a more detailed view, is based on a technology called Distributed Capability-Based Access Control (DCapBAC), which decouples the traditional XACML framework in two phases, one for receiving the authorization (represented by the receipt of the Capability Token) and a second one for accessing the information repository, where a Policy Enforcement Point (PEP) Proxy first validates the received Capability Token and in case of a positive answer it continues acting just as a mere intermediary with the information repository. Additionally, it interacts with other resource repositories placed in both DEH and BSE so that the access can always be controlled.

The authorization enabler depends on the resources repositories to be addressed, since they must incorporate the Capability Token to the corresponding queries so that the PEP Proxy can validate them.

8.2.2.3 DEH Discovery Management

The DEH Discovery Management module works together with the Resource Registry Management module presented in the previous subsection. In fact, for the most part, it piggybacks upon the information provided by that module in order to accomplish its task. Now, its task is to get requests for specific types of DEMETER enabled entities through the DEH interface (from users/stakeholders) and afterwards to query the Resource Management component (of the Resource Registry Management) in order to discover the resources matching the user request that the user is allowed to access; again the latter (access control) is being enforced by the data access functionalities of the Resource Registry making certain that when the list of available resources is returned as result of a user query, this list this does not contain any resources whose access policies would prohibit usage and discovery by the specific user.

Therefore, the main functionality of this component is to translate the data between the appropriate DEH Interface used by stakeholders in order to discover resources by placing the appropriate filters and the registry management component, and then to place the appropriate query through the relevant resource management interfaces. Once this information is returned to the module then it is processed and encoded appropriately to be shown through the DEH interface to the end user. It will be possible to change the ordering of the resources returned based on criteria such as price, quality, availability, platform or device used (if appropriate) just to mention a few. It will also be possible to keep track of the other queries made by the end user in order to provide resources which are compatible with the previous queries and resources sought previously by the same user in the same session.

Finally, in a second development stage for this component we will aim to store and reprocess the past queries submitted by users. This would allow us to inform users regarding new resources available which match their past queries for example. To accommodate such tasks, we will need to periodically rerun queries submitted by end users. This means that such information needs to be stored. If the resource registry cannot accommodate the storage of such information, then most likely this will require use of a simple database separate from the Resource Registry DB, inside the DEH Discovery Management module, in order to store this data. Any relevant resources which are then discovered will be logged, so that when the user reuses the DEH, this information will be available to them.

8.2.2.4 DEH Compatibility Checker

The Compatibility Checker component is an external module to validate if a resource is compatible with the DEMETER architecture. Compatibility Checker will check each new resource on several aspects of compatibility (before it can be registered and offered through the DEMETER HUB).

First, the compatibility checker will test the level (and completeness) of implementation of the defined DEMETER API.

- What services are implemented?
- Are mandatory parameters implemented?
- What optional parameters are implemented?
- Are responses correctly represented by codes?

Second, the compatibility checker will test the format of the services content. If requests and responses of the defined services accept payload in correct format and valid content; it should be

compatible with the AIM model offered by DEMETER. This test will need a defined set of demo requests and responses to assess correct content.

Third, the compatibility checker will test for correct semantic content. This test will need defined demo data set of different entities and objects. Tests will check for example:

- Is the timestamp represented by the correct datatype and with defined precision?
- Is the value of a measurement type (e.g., humidity) represented with defined datatype and with requested precision?
- Is the geometry of a feature represented with the same geometry type and defined in the same location regardless of the coordinate system?
- Is the identifier of a feature defined as unique?

All defined tests will contain defined weight to evaluate a measure of compatibility or conformance. Thresholds will be defined to evaluate tests results as accepted or not accepted.

8.3 *Process View*

The process view covers the internal dynamics aspect of a DEH with a focus on the runtime behaviour and describes processes inside it and interaction between them. For the purpose of representing the process view, UML Activity and Sequence diagrams will be used.

Activity diagrams will depict the activities flow and how they are coordinated inside DEH Components, while Sequence Diagrams will focus on requests and messages that the same components are sharing among them in order to execute the process.

8.3.1 **DEH Authentication & Authorization**

The diagram in Figure 34 depicts the sequence in the interaction between DYMER and components inside the Security Block from the moment the user enters his authentication credentials up to the moment where those credentials are validated alongside with access policies.

In the rest of this section, each diagram will imply that this part is passed, so each request from DYMER will contain X-AUTH-TOKEN.

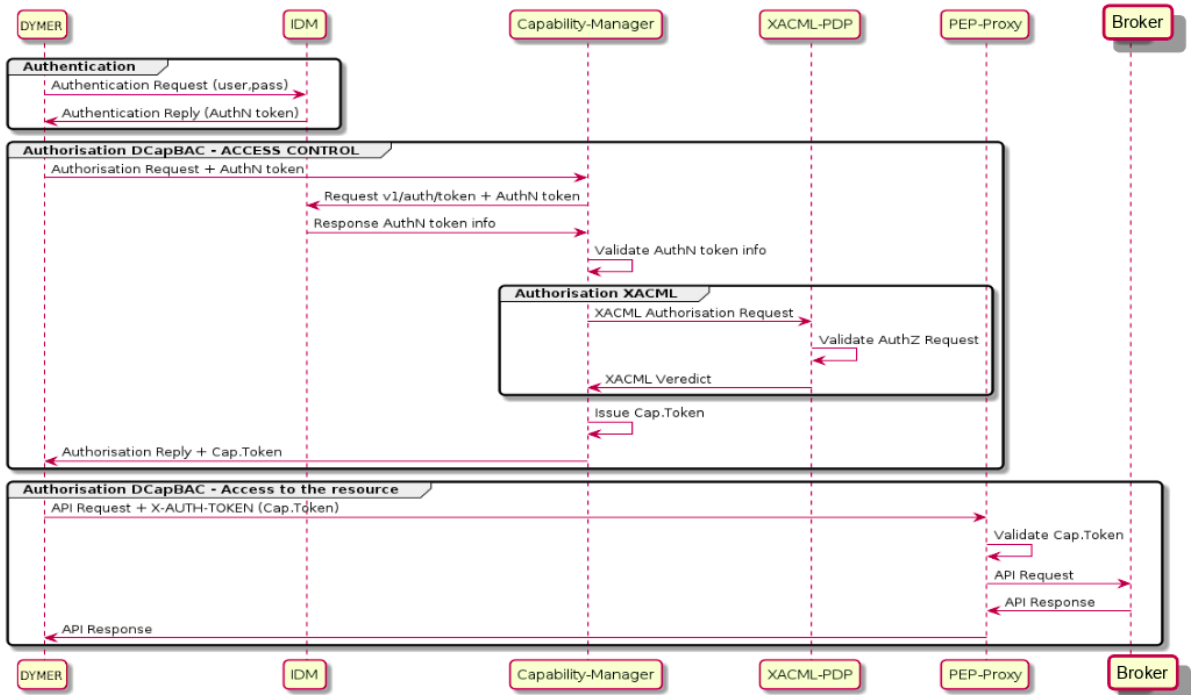


Figure 34: DEMETER Enabler Hub Authentication and Authorization sequence diagram

8.3.2 **DEH Resource Registry Management**

This subsection covers the interaction between internal DEH components in a process of Registering and Discovery Resources in DEH.

8.3.2.1 Activity Diagrams

Diagram in Figure 35 depicts the interaction between three DEH components that are involved in a process of registering a new resource:

- DYMER - DEH Dashboard component where developers are creating a new resource
- Resource Registry Management - component which manages all the processes related to the DEMETER resources
- Compatibility Checker - component which validates resource compatibility

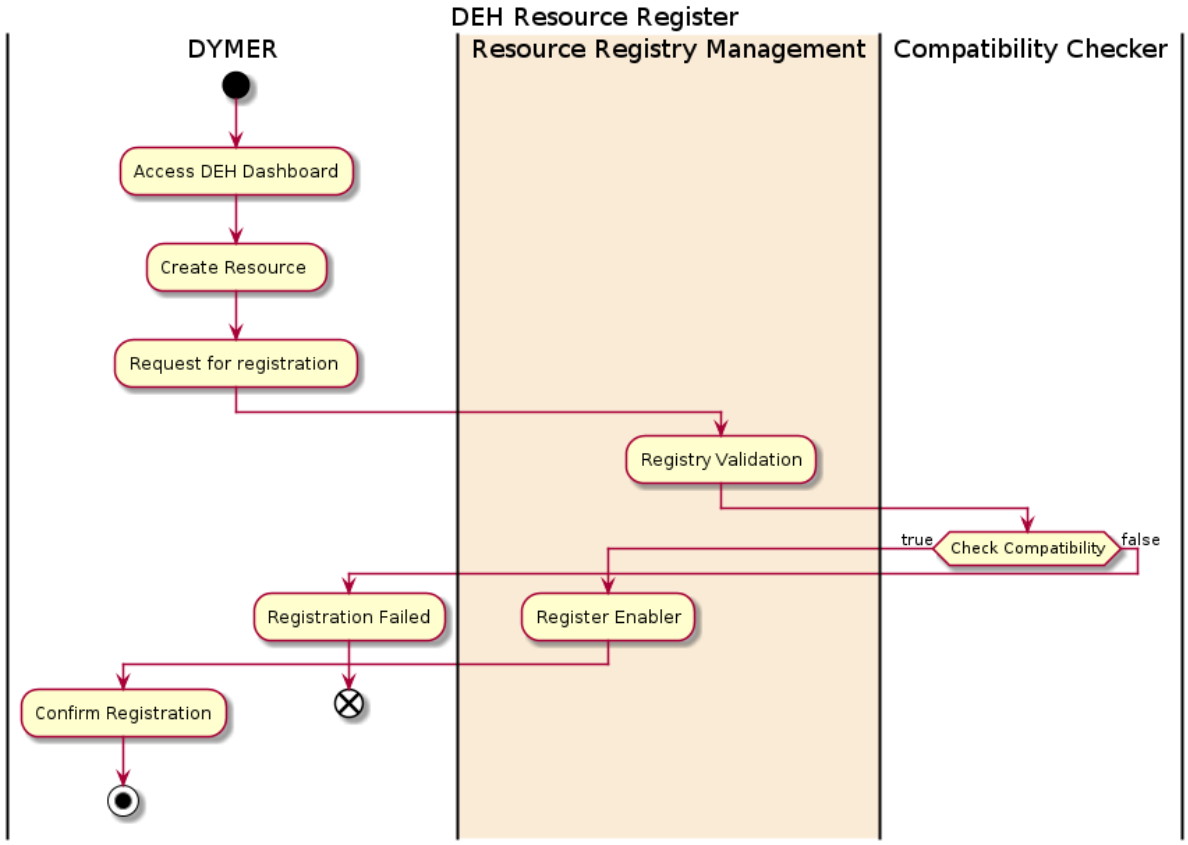


Figure 35: DEH Resource Registry activity diagram

To create a new resource in DEH, a DEMETER developer user must first authenticate and access his Dashboards (user GUI); subsequently you can send a request for registering a DEMETER Enabler. The Resource Registry management component prepares a resource for registration and requests a compatibility check. The Compatibility Checker component validates the registry. Based on validation, the registry will be saved as DEMETER enabler or not.

The diagram in Figure 36 depicts the interaction between four DEH components that are involved in a process of discovering resources:

- DYMER - DEH Dashboard component where user is searching for a resource
- Discovery Management - component which allows resource discovery
- Resource Access Control - component which checks the access policies of a user
- Resource Registry Management - component which manages all the processes related to the DEMETER resources

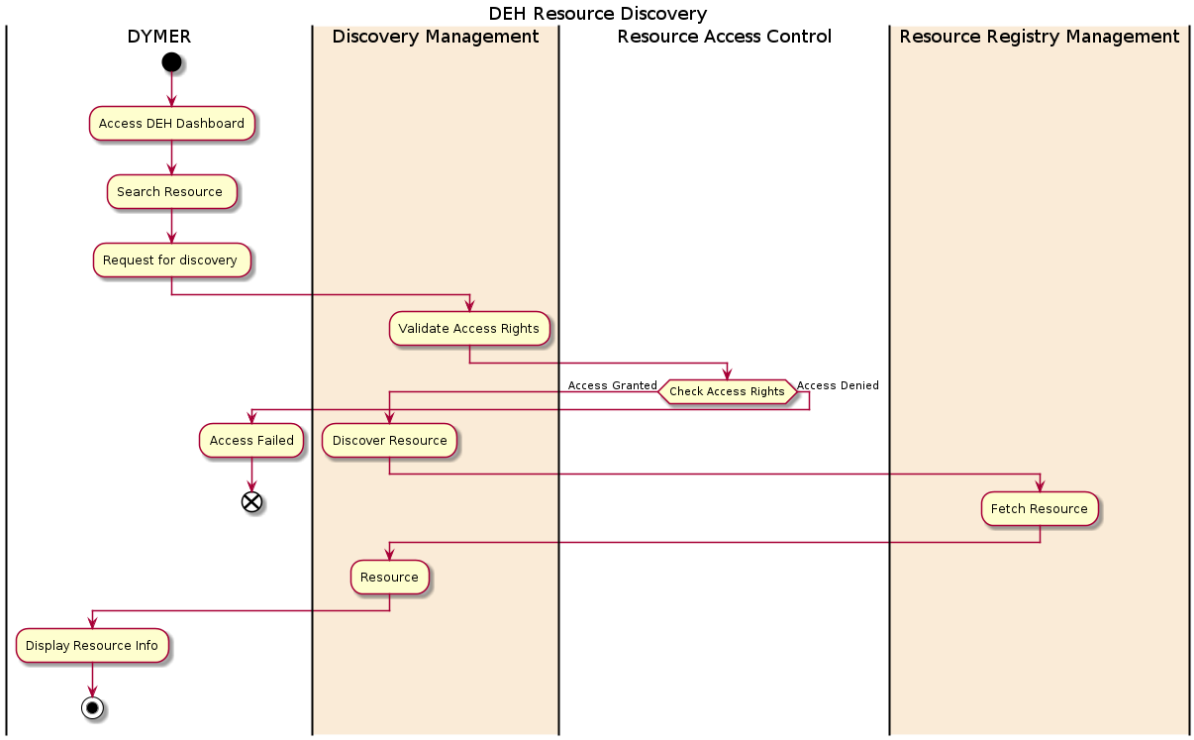


Figure 36: DEH Resource Discovery activity diagram

The Developer accesses the DEH dashboard to search for and request a resource. Discovery Management checks with Resource Access Control if the user has proper access policies. Depending on the result from the check against these access policies, the resource will be fetched, by sending a request to Resource Registry Management, and displayed to the user.

8.3.2.2 Sequence Diagrams

The diagram in Figure 37 depicts the sequence interaction between main components which are involved in a process of creating a new resource, as depicted in Figure 35 DEH Resource Registry Activity diagram:

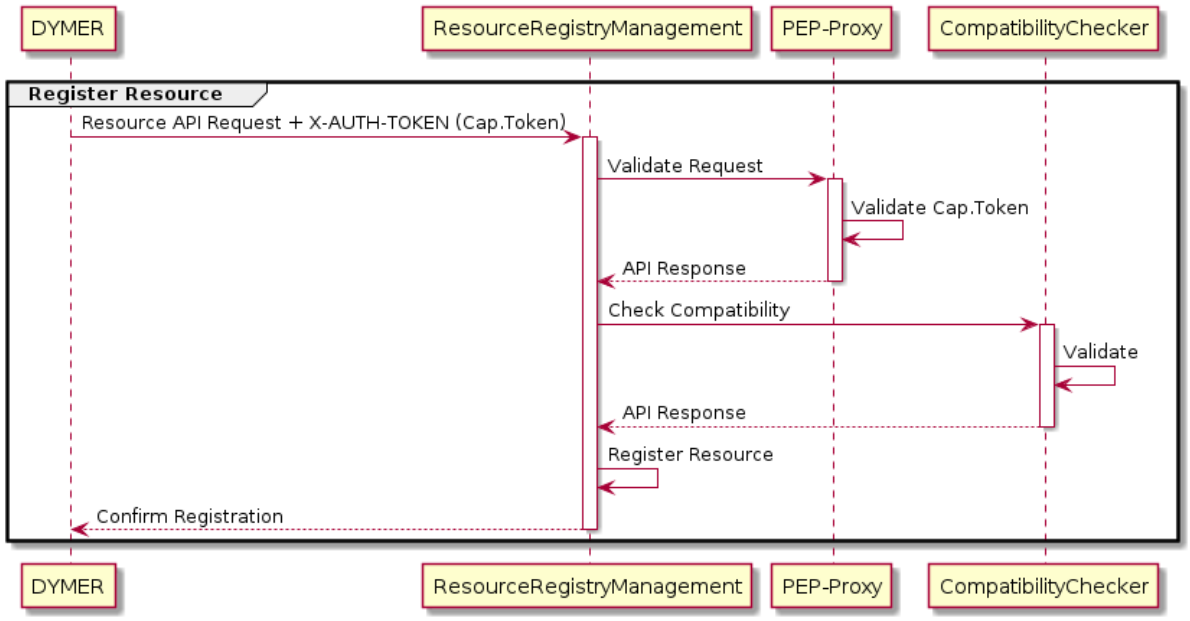


Figure 37: DEH Resource Registry sequence diagram

DYMER will send a request with X-AUTH-TOKEN in the header and the content related to the resource in a body to the DEH Resource Registry Management. The Resource Registry Management will contact the Resource Access Control to validate a request. Based on information in the header, the Resource Access Control will be able to determine if a user can register a new Resource or not and it will inform the DEH Resource Registry Management component. If a user is allowed, the Resource Registry Management component will check Compatibility of a new resource with the Compatibility Checker. Based on the information passed in the body of a request, the Compatibility Checker will be able to determine if a resource is compatible and it will inform the Registry Management component. If a resource is compatible, the Registry Management component will store a resource as a new Enabler in the DEMETER Resource Registry and send a response to DYMER with confirmation about storing.

Figure 38 shows the sequence diagram that depicts interaction between the main components which are involved in a process of discovering a resource depicted in Figure 36 DEH Resource Discovery Activity diagram:

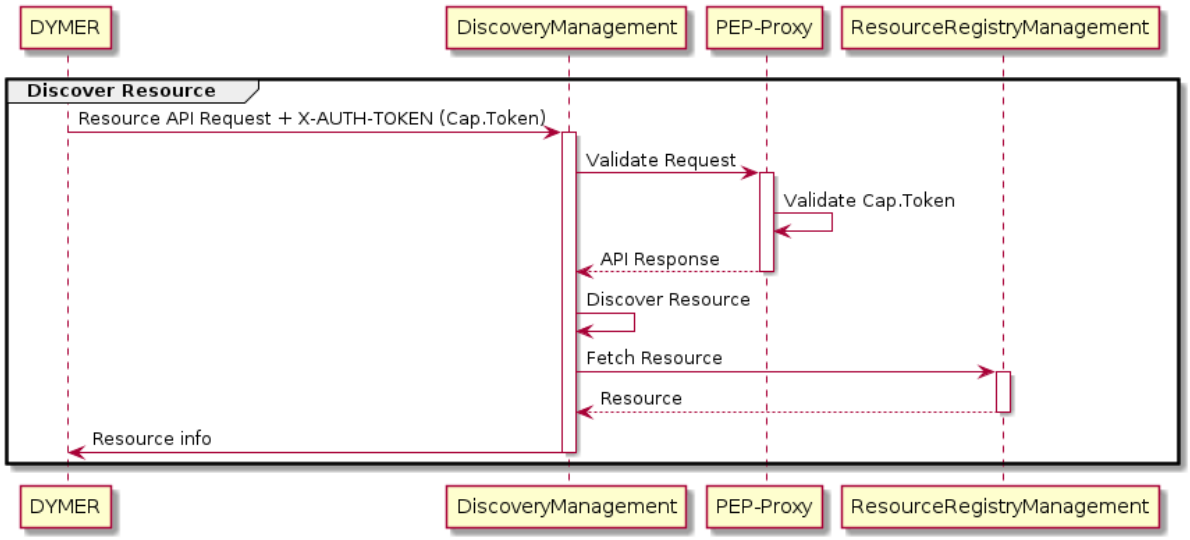


Figure 38: DEH Resource Discovery sequence diagram

The DYMER sends a request with the Capability Token in the X-AUTH-TOKEN header which will contain the resource id. The DEH Discovery Management module contacts the Resource Access Control to validate the request. Based on the Capability Token in the header, the Resource Access Control can determine if the request can access to the requested DEH resource or not and it informs the DEH Discovery Management component. If it is allowed, the DEH Discovery Management component can discover a resource, by calling the DEH Resource Registry Management component API, to fetch it. The DEH Resource Registry Management returns it then to the DEH Discovery Management, and from there it is passed to the DYMER application.

8.4 *Interfaces*

The data model presented in the following tables contains a preliminary list of properties that identify a resource within DEH; however, any additions to the models presented below are not excluded in terms of new properties or new types of resources that could emerge during the developments or integrate new requirements in terms of requesting data that DEH will have to support. Consequently, these definitions may change; the final version of DEH data model will be reported in D3.4.

8.4.1 **DEH Resource Data Model**

Table 7: DEH Resource Data Model

Name	*DEH Resource:* ***Component***, ***Device***, ***Service***, ***Dataset***, ***Platform***	
Property	**Type**	**Description**
UID	Alphanumeric	Resource unique identifier
Name	String	Resource name
Type	String	Resource type
Category	Arrays	Resource category
Description	String	Resource description
Endpoint	String	Resource endpoint
Status	String	Resource status
Version	String	Resource version

Maturity Level	Integer	Resource maturity level
Owner	String	Resource owner
Tags	Arrays	Resource tag
Attachment	Binary data	Resource attachment
Rating	Double	Resource rating
Localisation	Arrays	Resource localisation (geo-points)
Accessibility	Integer	Resource accessibility (0 = Public, 1=Private)
Last Update	Timestamp	Resource last update date
Dependencies	Arrays	Resource dependencies (with other resources)
Access controls policies	Arrays	Resource ACL
URL	String	URL for downloading/streaming data or entity
Billing information	Arrays	Resource billing information

Table 8: DEH User Data Model

Name	*DEH User*	
Property	**Type**	**Description**
Username	String	User username
Password	String	User password
First name	String	User first name
Last name	String	User last name
Email address	String	User email address
Phone number	Double	User phone number
Country	String	User country
Organisation/company	String	User organisation/company
Sectors of interest	String	User sectors of interest
Category/Type	String	User type {Developers, Farmer, Advisor}

8.4.2 **Description of APIs**

The DEH API described in the templates below represent a first version (v1) of the DEH API software stack that will be produced during the project. These services or preliminary set of APIs could be integrated with other services that will become necessary for changed conditions or for application needs or simply updated and improved the existing ones. Consequently, these refinement and new API definitions may change; the final version of the DEH API will be reported in D3.4.

8.4.2.1 DEH Dashboard (DYMER Core API)

Title	*Get entities*
URL:	
/api/entities/api/v1/entity/_search	
Headers	
enctype: "multipart/form-data	
Method	
POST	

URL Params	
Required:	
N/A	N/A
Optional:	
N/A	N/A
Data Params	
Required:	

{
"bool": {
"must": [
{
"term": {
"category": "open"
}
}
]
}
}
}
	query: json object with the desired query

Optional:	
N/A	*N/A*
Success response	

200
Content: {
"success": true,
"message": "",
"data": [ ],
"extraData": {}
}
	success: true if there no errors
	message:message form the server eg. Resouce List or Empty List
	data: json array of objects
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Error response	

404
Content: {
"success": false,
"message": "",
"data": [ ],
"extraData": {}
}
	success: false if there is an errors
	message:message form the server eg. Resouce List or Empty List
	data: empty json array
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

Title	*Get File by ID*
URL:	
api/entities/api/v1/entity/content/:fileid	
Headers	
mimeType: "text/html", contentType: "text/html"	
Method	
GET	
URL Params	
Required:	
fileid = [integer]	*ID of the file contained in the json object*

Optional:	
N/A	N/A
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { file }	*Response contains file*
Error response	
404	*Resource not found*
Sample call	
N/A	
Notes	
N/A	

Title	*Create Entity*
URL:	
/api/entities/api/v1/entity/	
Headers	
enctype: "multipart/form-data;"	
Method	
POST	
URL Params	
Required:	
N/A	N/A
Optional:	
N/A	N/A
Data Params	
Required:	

{
"instance":{
"index":"demeterproduct",
"type":" demeterproduct "
},
"data":{
….
"properties":{}
}
}
	instance: json object with define the type and index of entity
	
	---:key:value/jsonobject/jsonarray that define details of the entity
	
	properties: json object with define the properties of the entity

Optional:	
N/A	*N/A*
Success response	

200
Content: {
"success": true,
"message": "",
"data": [ ],
"extraData": {}
}
	success: true if there no errors
	message:message form the server eg. "Entity successful created"
	data: json array of objects
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Error response	

404
Content: {
"success": false,
"message": "",
"data": [ ],
"extraData": {}
}
	success: false if there is an errors
	message:message form the server eg. "Entity could not be created"
	data: empty json array
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

Title	*Edit Entity*
URL:	
/api/entities/api/v1/entity/:id	
Headers	
enctype: "multipart/form-data;"	
Method	
PUT	
URL Params	
Required:	
id=[integer]	*ID of the entity to update*
Optional:	
N/A	N/A
Data Params	
Required:	

{
"instance":{
"index":"demeterproduct",
"type":" demeterproduct "
},
"data":{
….
"properties":{}
}
}
	instance: json object with define the type and index of entity
	
	---:key:value/jsonobject/jsonarray that define details of the entity
	
	properties: json object with define the properties of the entity

Optional:	
N/A	*N/A*
Success response	

200
Content: {
"success": true,
"message": "",
"data": [ ],
"extraData": {}
}
	success: true if there no errors
	message:message form the server eg. "Entity successful updated"
	data: json array of objects
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Error response	
404 Content: { "success": false, "message": "", "data": [ ],	*success: false if there is an error* *message:message form the server eg. "Entity could not be updated"* *data: empty json array*

"extraData": {} }	*extraData: optional json object which contains extra informazions eg. Logs, data etc.*
Sample call	
N/A	
Notes	
N/A	

Title	*Delete Entity*
URL:	
/api/entities/api/v1/entity/:id	
Headers	
enctype: "multipart/form-data;"	
Method	
PUT	
URL Params	
Required:	
id=[integer]	*ID of the entity to update*
Optional:	
N/A	N/A
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	

200
Content: {
"success": true,
"message": "",
"data": [ ],
"extraData": {}
}
	success: true if there no errors
	message:message form the server eg. "Entity successful deleted"
	data: json array of objects
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Error response	

404
Content: {
"success": false,
"message": "",
"data": [ ],
"extraData": {}
}
	success: false if there is an errors
	message: message from the server eg."Entity could not be deleted"
	data: empty json array
	extraData: optional json object which contains extra informazions eg. Logs, data etc.

Sample call	
N/A	
Notes	
N/A	

8.4.2.2 Resource Registry Management

Title	*Get Resource by UID*
URL:	
v1/resource/{uid}	

Method	
GET	
URL Params	
Required:	
uid= alphanumeric]	*Resource UID*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*All information related to resource with given UID*
Error response	
401, 403, 404, 500, 503, 504	*Unauthorized, Access forbidden, Resource not found, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Get Resources by Type*
URL:	
/api/v1/resources/{type}	
Method	
GET	
URL Params	
Required:	
type=[String]	*Resource type*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*List of resources by type*
Error response	
401, 403, 404, 500, 503, 504	*Unauthorized, Access forbidden, Resource not found, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Get resources by category*
URL:	
/api/v1/resources/{category}	
Method	
GET	
URL Params	
Required:	
category=[String]	*Name of resource category*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*List of resources by category*
Error response	
401, 403, 404, 500, 503, 504	*Unauthorized, Access forbidden, Resource not found, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Get Resources by Tags*
URL:	
/api/v1/resources/{tags}	
Method	
GET	
URL Params	
Required:	
tags=[Arrays]	*Resource tags*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*List of resources by tags*
Error response	

401, 403, 404, 500, 503, 504	*Unauthorized, Access forbidden, Resource not found, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Get resources by text search*
URL:	
/api/v1/resources/	
Method	
GET	
URL Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Data Params	
Required:	
{text_input=string}	*Resource that mach with the text input param*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*List of resources by text search*
Error response	
401, 403, 404, 500, 503, 504	*Unauthorized, Access forbidden, Resource not found, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Create a new resource*
URL:	
api/v1/resource/	
Method	
POST	
URL Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Data Params	
Required:	
name=[String]	*resource name*

description=[String]	*resource description*
owner=[String]	*resource owner*
Optional:	
uid=[Alphanumeric]	*resource uid*
status=[String]	*resource status*
category=[Arrays]	*resource categories*
type=[String]	*resource type*
endpoint=[String]	*resource endpoint*
version=[String]	*resource version*
maturitylevel=[Integer]	*resource maturity level*
localisation=[Arrays]	*resource localisation geo-points*
url=[String]	*URL for downloading/streaming data or entity*
accessibility=[Integer]	*resource accessibility*
access_control_policies=[Array of policies]	policies regarding who can access the resource (e.g. excluding specific user types)
Success response	
200 Content: { }	*N/A*
Error response	
401, 403, 500, 503, 504	*Unauthorized, Access forbidden, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Update an existing resource*
URL:	
api/v1/resource/{uid}	
Method	
PUT	
URL Params	
Required:	
uid=[String]	*resource uid*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
description=[String]	*resource description*
endpoint[String]	*Resource endpoint*
status=[String]	*resource status*
category=[Arrays]	*resource categories*
maturitylevel=[Integer]	*resource maturity level*
localisation=[Arrays]	*resource localisation geo-points*
tags=[Arrays]	*Resource tags*
url=[String]	*URL for downloading/streaming data or entity*
accessibility=[Integer]	*resource accessibility*
attachment=[byte]	*resource attachment*
rating=[Double]	*resource rating*

access_control_policies=[Arrays]	
dependencies=[Arrays]	*resource dependencies*
billilng_information=[Arrays]	*resource billing information*
Success response	
200 Content: { }	resource updated
Error response	
401, 403, 500, 503, 504	*Unauthorized, Access forbidden, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

Title	*Delete resource*
URL:	
api/v1/resource/{uid}	
Method	
DELETE	
URL Params	
Required:	
uid=[integer]	*resource uid*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*resource deleted*
Error response	
401, 403, 500, 503, 504	*Unauthorized, Access forbidden, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

8.4.2.3 Resource Discovery

Building upon the previous interfaces of the resource registry we will also need (at least) the following interface for accessing resources from the registry that match multiple criteria (including and enforcing access control policies):

Title	*Get Resources matching multiple criteria*
URL:	

/api/v1/resources/advancedSearch?<query with the data, e.g. type etc.>	
Headers	
enctype: "multipart/form-data	
Method	
GET	
URL Params	
Required:	
type=[String]	*Resource type*
category=[String]	*Name of resource category*
tags=[Arrays]	*Resource tags*
text_input=[String]	*Resources that match with the text input param*
access_control_policies=[Arrays]	*Input for access control policies*
Optional:	
N/A	*N/A*
Data Params	
Required:	
N/A	*N/A*
Optional:	
N/A	*N/A*
Success response	
200 Content: { }	*List of resources by type*
Error response	
401, 403, 404, 500, 503, 504	*Unauthorized, Access forbidden, Resource not found, Internal Server Error, Service Unavailable, Gateway Timeout*
Sample call	
N/A	
Notes	
N/A	

8.5 *Technologies and implementation details*

This section summarizes used technologies, tools and frameworks linked to the implementation of DEH.

Table 9 summarizes resources linked to the DEH Dashboard:

Table 9: DEH Dashboard linked resources

AngularJS	https://docs.angularjs.org/api
NodeJS	https://nodejs.org/en/docs
ElasticSearch	https://www.elastic.co/guide/index.html
MongoDB	https://docs.mongodb.com
Express.js	https://expressjs.com/en/guide/routing.html

DEH Dashboard has two parts, the Front-End part developed in AngularJS which is SPA (Single Page Application) framework and the back-end part developed in Node.js, which is JavaScript runtime environment that executes JavaScript code outside a web browser, where Express.js is used for

routing. For the purpose of storing templates, services and forms MongoDB which is a NoSQL document-oriented database is used. Elasticsearch is used for the purpose of storing entities, their indexing, and fast searching.

Table 10 summarizes resources linked to the DEH Resource Registry Management:

Table 10: DEH Resource Registry Management linked resources

Spring Boot	https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
MongoDB	https://docs.mongodb.com/
Spring Data MongoDB	https://docs.spring.io/spring-data/mongodb/docs/current/reference/html
Swagger	https://swagger.io/docs/
Lombok	https://projectlombok.org/features/all
Apache Maven	https://maven.apache.org/guides/index.html

The DEH Resource Registry Management is developed using Spring Boot which is an industry-standard Java-based Framework, with Maven as a build automation tool. For the purpose of storing the data related to resources MongoDB which is a NoSQL document-oriented database. As additional modules that are used within Spring eco-system, we can mention Spring Data MongoDB which is used to make it easier to manage data in MongoDB. Lombok is used to reduce the boilerplate code related to Entities, while Swagger is used for the purpose of documenting and testing REST APIs.

As the DEH Discovery Management, the design, and operations of which are briefly described in section 8.2.2, is essentially based on data provided by the DEH Resource Registry Management components. This essentially means that the same technologies used in the aforementioned module will be sufficient for the development of the DEH Discovery Management as well. These cover all the key requirements for resource discovery as laid out by the requirements analysis. For the second phase of the development of this component we may need to develop some additional support tools which would probably require the use of a simple database in this tools separate from the Resource Registry DB (easily covered by MongoDB) in order, e.g., to store and reprocess the past queries submitted by users.

Table 11 summarizes resources linked to the DEH Compatibility Checker.

Table 11: Compatibility Checker linked resources

Django REST framework	https://www.django-rest-framework.org/
Django	https://www.djangoproject.com/
PostgreSQL	https://www.postgresql.org/
Swagger	https://swagger.io/docs/

9 Core Enablers for Integration

9.1 *Functional Interoperability Enabler*

9.1.1 Text information – metadata

9.1.1.1 Functionality description

Functional Interoperability Core Enabler is the client-side of the Brokerage Service Environment. This Enabler provides all the services of the BSE to the rest of the Enablers (Core and Advanced) and to the Consumer's application. It serves as a wrapper for the Registration, Discovery, and Provisioning services offered by the BSE.

9.1.1.2 Interaction with other Enablers

This Enabler offers an HTTP API to any other Enabler (Core and Advanced) that needs to make use of the provided services.

9.1.1.3 Dependencies on other Core/Advanced Enablers

This Enabler depends on the Security Enabler (also a Core DEMETER Enabler) as it requires the access credentials (authentication/authorization) to successfully communicate with the BSE. It also depends on the Semantic Interoperability Core Enabler (WP2) and the Agricultural Information Model (AIM) related functionality that this Enabler offers.

9.1.1.4 Deployment considerations

The container image of this Enabler will be available via DEMETER's Image Registry (described in section 10). It will be freely available to all DEMETER consortium and the requirements are minimal, i.e., OS capable to run docker containers, docker service up and running.

9.1.2 Technical description

This information formally describes features/characteristics of this Enabler

9.1.2.1 API and Data model

Table 12: Functional Interoperability Enabler Data Model Information

Name	Functional Interoperability Core Enabler data model	
Property	Type	Description
timestamp	Timestamp	The transaction timestamp
resource_id	String	The resource unique id
resource_name	String	The resource name
resource_access_info	JSON	Information on how to access the resource (e.g., port, protocol, URL, etc)
resource_metadata	JSON	Metadata information for the resource (e.g., vendor, version, etc)
resource_validation_info	JSON	Information on how to validate the resource (e.g., validation endpoints, expected responses, etc)

resource_dependencies	Array	Dependencies on other resources
resource_usage_info	JSON	Information on the usage of the resource (e.g., accepted request rate, restrictions on concurrent consumers, etc)
resource_tags	Array	Tags for discoverability
start_time	Timestamp	Start time (e.g., the start time in a resource provisioning request)
end_time	Timestamp	End time (e.g., the end time in a resource provisioning request)
user_id	String	The provider/consumer unique identifier
provision_request_info	JSON	Information on the resource provisioning request (e.g., requested duration, rate, number of devices, number of users, etc)
provision_access_info	JSON	Information on the provisioning (e.g., duration of access, rate of access, restrictions on concurrent connections, etc)

Developers are strongly advised to adopt Swagger for online documentation of the developed APIs. Swagger details for the online documentation will also be provided.

Title	Register resource to BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http://functionalinteroperability/api/v1/resource	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
timestamp	The timestamp of registration
user_id	The unique identifier of the provider
resource_name	The name of the resource to be registered
resource_access_info	The access info of the resource
resource_metadata	The metadata of the resource
resource_validation_info	The validation info of the resource
resource_dependencies	The dependencies of the resource

resource_usage_info	The usage information of the resource
resource_tags	The tags for the resource
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {resource_id}	Request was successful
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Modify registered resource to BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http:// functionalinteroperability /api/v1/resource	
Method This field holds the type of the Method used	
PUT	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identified of the provided
resource_id	The unique identifier of the resource
Optional:	
resource_name	The name of the resource
resource_access_info	The access info of the resource
resource_metadata	The metadata of the resource
resource_validation_info	The validation info of the resource
resource_dependencies	The dependencies of the resource
resource_usage_info	The usage information of the resource
resource_tags	The tags for the resource
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: { }	Resource was successfully modified

Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Remove registered resource from BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http:// functionalinteroperability /api/v1/resource	
Method This field holds the type of the Method used	
DELETE	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the provider
resource_id	The unique identifier of the resource
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: { }	Resource was successfully deleted
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Discover registered resource from BSE
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	

http:// functionalinteroperability /api/v1/resource	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request
Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the consumer
Optional:	
resource_id	The unique identifier of the resource
resource_name	The name of the resource
resource_metadata	The metadata of the resource
resource_tags	The tags for the resource
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: [resource_id: { resource_name: String, resource_metadata: JSON, resource_validation_info: JSON, resource_dependencies: [String], resource_usage_info: JSON, resource_tags: [String] }]	An array of resource objects discovered
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

Title	Provision registered resource
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
http:// functionalinteroperability /api/v1/provision	
Method This field holds the type of the Method used	
GET	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
Content-Type=application/json	Header for json request

Optional:	
Data Params This field holds the body payload of a request.	
Required:	
user_id	The unique identifier of the consumer
resource_id	The unique identifier of the resource
Optional:	
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
200 Content: {resource_access_info}	Provisioning and access information
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	Not found
403	Not authorized
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
N/A	
Notes This field holds any additional helpful info related to this endpoint.	

9.1.2.2 Use cases / Data flow

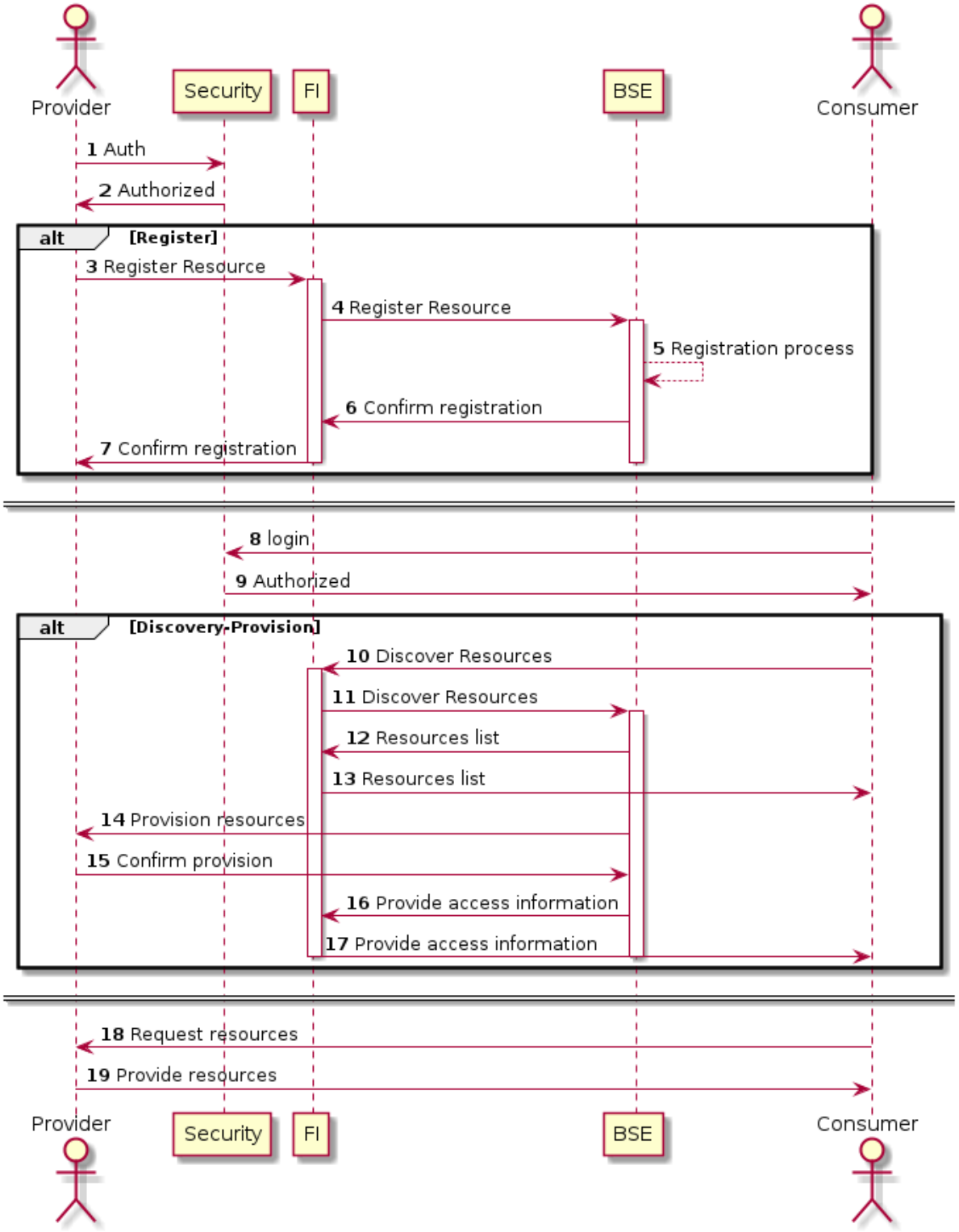


Figure 39: Functionality Enabler (FI) Sequence diagram

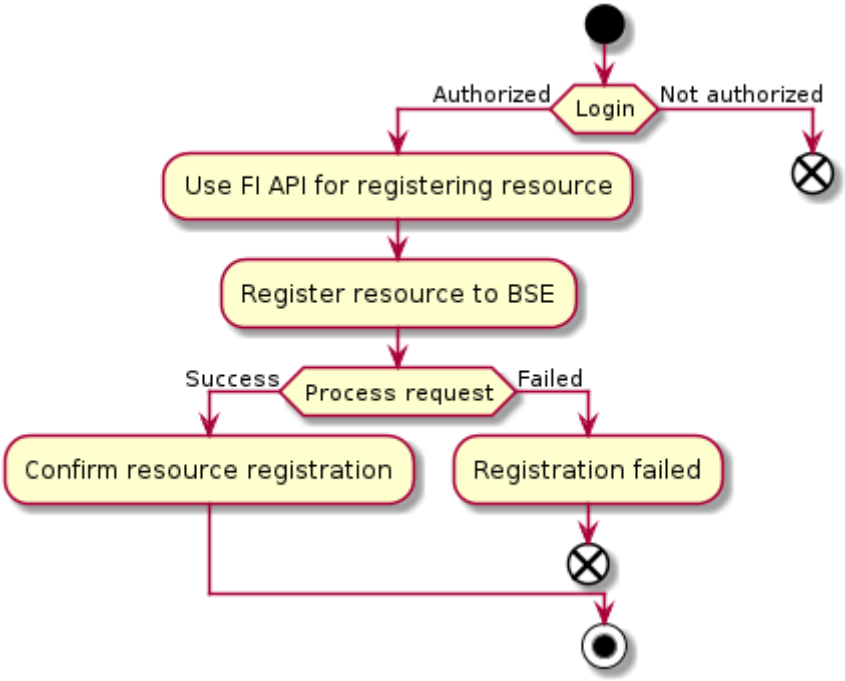


Figure 40: FI Activity diagram - Provider

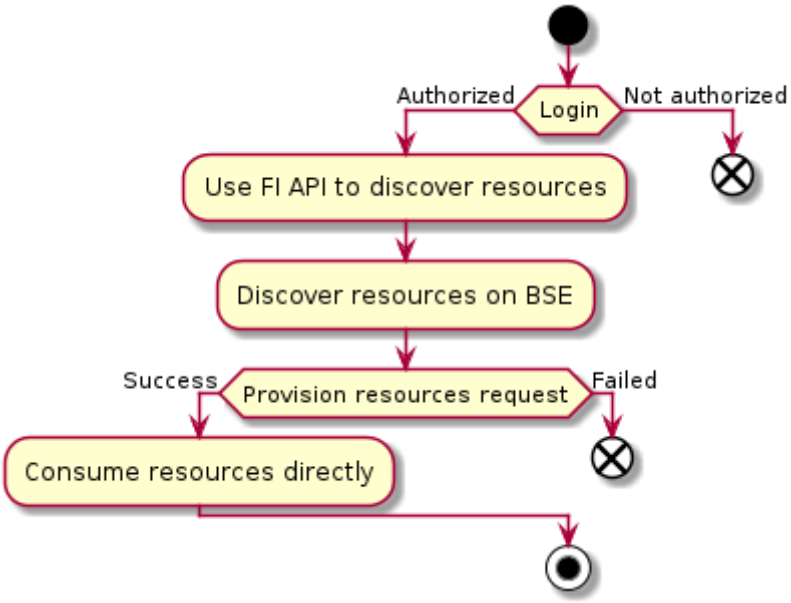


Figure 41: FI Activity diagram - Consumer

9.1.2.3 Deployment

In this section, we describe the deployment process for the FI enabler using a Docker-compose script and the deployment execution commands.

```
version: '3.4'
services:
  functional_int_enabler:
     image: demeter-project.eu/registry/:functional_int_enabler:0.0.1
```

```
    container_name: fi_enabler
    restart: always
    environment:
       - SEC_URL = "https://path/to/security/enabler"
       - SEC_PORT = 0000
       - SI_URL = "https://path/to/semantic/interoperability/enabler"
       - SI_PORT = 0000
       - BSE_URL = "https://path/to/BSE"
       - BSE_PORT = 0000
       - SSL_PATH = "path/to/SSL/certificate"
   depends_on:
      - Security_Enabler
     - Semantic_Interoperability_Enabler
   ports:
     - "8080:8080"
```

Deploy by:

```
sudo docker-compose up -d
```

9.1.2.4 Configuration Parameters

Configuration parameter	Value	Type	Description
SEC_URL	Defined by user	String	Security Enabler URL
SEC_PORT	Defined by user	Number	Security Enabler port
SI_URL	Defined by user	String	Semantic Interoperability URL
SI_PORT	Defined by user	String	Semantic Interoperability port
BSE_URL	Defined by user	String	BSE URL
BSE_PORT	Defined by user	Number	BSE port
SSL_PATH	Defined by user	String	The path to SSL certificate

9.2 ***Security Enabler***

9.2.1 **Authentication Security Enabler**

9.2.1.1 Functionality description

The Security Authentication Enabler library provides to the DEMETER components and the pilots developers an abstract way to access to the Authentication OAuth 2.0 functionalities exposed by the DEMETER Authentication component REST API.

This library provides the following functions:

- Authentication by username and password
- Refresh authentication
- Revoke authentication token

9.2.1.2 Interaction with other Enablers

The Security Authentication Enabler may need to interact with the Communication and Networking Enabler to obtain a secured communication channel to perform the authentication functionalities.

This enabler will also provide to the Security Authorisation Enabler(s) the authentication token needed to perform authorization functionalities.

9.2.1.3 Dependencies on other Core/Advanced Enablers

The functionalities provided by the security enablers (e.g. https communication, authentication and authorization tokens) will be used by the other Core/Advanced Enablers and other DEMETER components in order to obtain a secured communication channel and get access right to DEMETER resources. Therefore, the security enablers do not have any dependencies with other Enablers or DEMETER components.

9.2.1.4 Deployment/Development considerations

The authentication security enabler will be provided as a dynamic library, initially for both Windows and Linux Operating Systems.

This dynamic library can be used in different programming languages and frameworks.

9.2.1.5 Technical description

This information formally describes features/characteristics of the authentication Enabler.

Functions and Data model

The following functions are provided by this dynamic library in order to obtain, refresh and revoke authentication tokens:

Title	*Create token with Username and Password*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
get_authentication_token(username, password)	
Output *This field holds the type of the output expected*	
Authentication token (string) and expiration (time/date)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
username=[string]	*String with the username to log in*
Required:	
password=[string]	*String with the password to log in*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
Authentication_Token:04c5b070-4292-4b3f-911b-Authentication_Token_expires_at:"2018-03-20T15:05:35.697Z"	*Authentication token and its expiration date to be used with following authentication/authorisation functions.*
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
400, "Invalid client: client is invalid"	*There has been a time out event while connecting to Keyrock Server*
400, "Invalid grant: user credentials are invalid"	*The username or password provided doesn´t match any registered user in Keyrock*
Sample call *This field holds a possible sample call to the described function*	
get_authentication_token ("user.example@example.com", "password1234")	

Notes *This field holds any additional helpful info related to the function described.*

Title	*Refresh token*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
refresh_authentication_token(authentication_token)	
Output *This field holds the type of the output expected*	
Authentication token (string) and expiration (time/date)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
authentication =[string]	*String with the authentication token*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
Authentication_Token: 65c6b870-3535-6b4f-345b-34a345f3ac7f Authentication_Token_expires_at:"2018-03-20T15:05:35.697Z"	*New authentication token and its expiration date to be used with following authentication/authorisation functions.*
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
400, "Invalid grant: refresh token is no longer valid"	*The token provided is not longer valid, therefore, a new authentication token is not provided.*
Sample call *This field holds a possible sample call to the described function*	
refresh_authentication_token(65c6b870-3535-6b4f-345b-34a345f3ac7f)	
Notes *This field holds any additional helpful info related to the function described.*	

Title	*Revoke token*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
revoke_authentication_token(authentication_token)	
Output *This field holds the type of the output expected*	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
authentication =[string]	*String with the authentication token*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
0	*Success response for token deletion.*
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
400, "Invalid grant: refresh token is no longer valid"	*The token provided is no longer valid.*
Sample call *This field holds a possible sample call to the described function*	
revoke_authentication_token ("65c6b870-3535-6b4f-345b-34a345f3ac7f")	
Notes *This field holds any additional helpful info related to the function described.*	

Use cases / Data flow

The following figure, depicts the sequence diagrams for *get_authentication_token*, *refresh_authentication_token* and *revoke_authentication_token* functions.

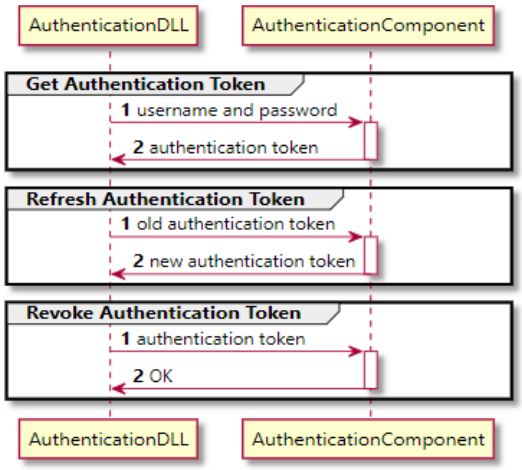


Figure 42: Authentication function sequence diagrams

The functions obtain the parameters and send it to the Authentication Component endpoint, where is processed and a response is provided, either with a new authentication token (*authentication_token*, *refresh_authentication_token*) or with the confirmation that an authentication token has been revoked (*revoke_authentication_token*).

Deployment

The library needs to be imported in the programming language of choice, and the function imported. The following examples show how to import them for several well-known and widely used programming languages such as Python, Java and C#:

Python:

```
from demeter_authentication import login_with_password, login_with_client_credentail, refresh_token

authentication_token, expire_at = login_with_password(“user1@example.com”,”password123”)
```

Java:

```
import static demeter_authentication.*;

authentication_token, expire_at = login_with_password(“user1@example.com”,”password123”)
```

C#:

```
using demeter_authentication;

authentication_token, expire_at = login_with_password(“user1@example.com”,”password123”)
```

Configuration Parameters

The following configurations parameters are need for the library to access to the DEMETER Authentication Component:

Configuration parameter	Value	Type	Description
KEYROCK_URL	URL	String	Keyrock Endpoint

9.2.2 **Authorisation Security Enabler**

9.2.2.1 Functionality description

The authorization enabler provides a solution for controlling the access to the resources stored in an information repository. It is based on a technology called Distributed Capability-Based Access Control, which basically decouples the traditional XACML framework, into two phases: one for receiving the authorization, which is represented by the receipt of an authorisation token called Capability Token, and a second one for accessing the information repository where basically, the user/service inserts the previous Capability Token in the corresponding query so that a Policy Enforcement Point Proxy (PEP_Proxy) could check if the query matches the content of the Capability Token. In case of a positive answer, the PEP_Proxy acts as a mere intermediary between the user/service and the information repository.

9.2.2.2 Interaction with other Enablers

This enabler interacts with the authentication enabler. Before performing the authorisation process, the authentication one must be carried out. After this authentication phase, an authentication token is generated, and this token must be present in the authorisation requests. This way, the authorisation enabler interacts with the authentication enabler in order to validate this token.

Additionally, this enabler interacts with other resource repositories placed in both BSE and DEH so that the access to the different resource repositories can be controlled. So far, the current implementation depends on NGSI or NGSI-LD resource repositories.

9.2.2.3 Dependencies on other Core/Advanced Enablers

The authorisation enabler depends on the resource repository to be addressed by user/services, since they must incorporate the Capability Token to the corresponding queries so that the PEP_Proxy would be able to validate them.

9.2.2.4 Deployment/Development considerations

The authorisation enabler comprises different sub-components, nevertheless, only the endpoint for the Capability Manager is provided to the other components. For this reason, it can be accessed by following a specific REST API. Additionally, a java library (jar) has been developed to make it easier to interact with the corresponding servers. This library, since uses JAVA can be run over different OSs.

9.2.2.5 Technical description

This information formally describes features/characteristics of an Enabler.

Functions and Data model

Data model used: each of the parameters received by this function are strings.

- Function details:
  - Name: generateCapabilityToken(“authtoken”,”subject”,”resource”,”action”)
  - Expected output: CapabilityToken. A signed JSON document.
  - Error messages: Error connecting to the Authorisation server.

Data models used by the functions/methods shall be described in tables:

Table 13: Authorisation Enabler Data Model Information

Name	*Authentication Enabler Data Model*	
Property	**Type**	**Description**
Authtoken	String	The token obtained from the Identity Management
Subject	String	The subject of the authorisation query
Resource	String	The resource intended to access
action	String	The operation mode: GET, POST, PUT, PATCH or DELETE

Title	*Generate Capability Token*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
generateCapabilityToken *(authtoken,subject,resource,action)*	
Output *This field holds the type of the output expected*	
Authorisation token (json document)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
authtoken=[alphanumeric]	*Alphanumeric string with the authentication token*
Required:	
subject=[alphanumeric]	*Alphanumeric string with the subject of the authorisation request*
Required:	
Resource=[alphanumeric]	*Alphanumeric string identifying the resource to be accessed*
Required:	
Action=[alphanumeric]	*Alphanumeric string corresponding to the operation to be performed: GET, POST, PUT, PATCH or DELETE*
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
Authorisation token: { “id”: “7g3vfT_q9vTL2aQ4”, “ii”: 1415174237, “is”: “issuer@um.es”, “su”: “zNwS5FetB4rwzSKsWwSBAxm5wDa=JgLjHU8zSnmeSFQgSG9HhdsJrE8=”, “de”: “coap://sensortemp.floor1.computersciencefaculty.um.es”, “si”: “SbUudG4zuXswFBxDeHB87N6t9hR=PBQqCN3gpu7nSkuPzDk7kaR3dq1=”, “ar”: [{ “ac”: “GET”, “re”: “temperature”, “f”: 1, “co”: [{	*Authorisation token.*

“t”: 5,
“v”: 25,
“u”: “Cel”,
},
{
“t”: 6,
“v”: 20,
“u”: “Cel”,
}
]
}
],
“nb”: 1415174237,
“na”: 1415175381
}	

Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
-1, “connection timeout”	*There has been a time out event while connecting to Authorisation Server*
Sample call *This field holds a possible sample call to the described function*	
generateCapabilityToken(“04c5b070-4292-4b3f-911b-“,jamartinez@odins.es,”ngsi-ld/v1/entities”,”GET”)	
Notes *This field holds any additional helpful info related to the function described.*	

Use cases / Data flow

Authorisation DCapBAK access control flow is presented in Figure 43 below.

UML Sequence diagram(s)

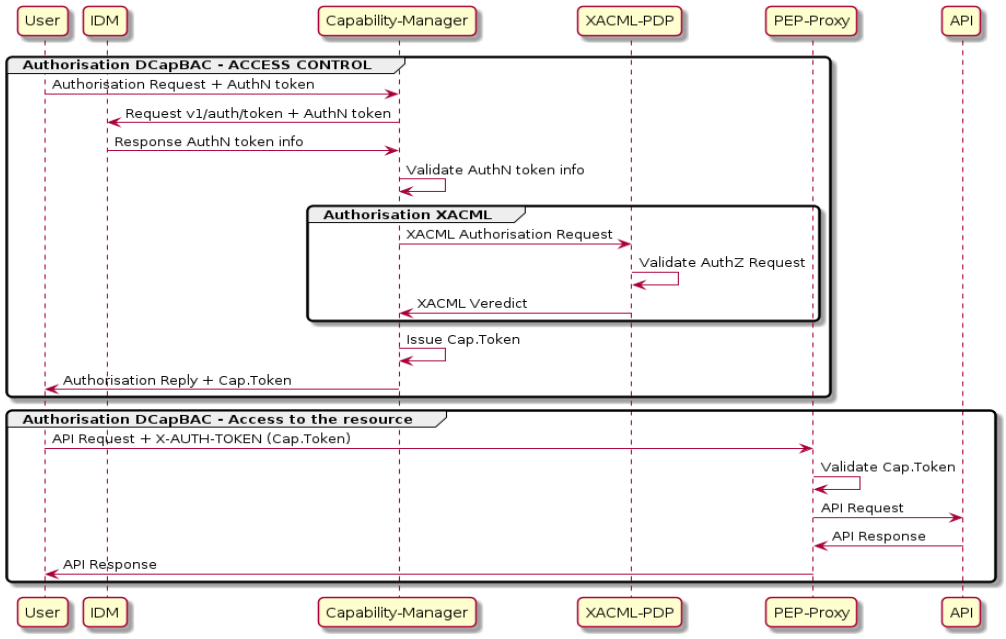


Figure 43: Authorisation DCapBAC access control sequence diagram

Deployment

Technically describe the deployment process for the enabler: examples of how to import the library for different programming languages.

```
i.e java:
import demeter_authorisation;
…
capToken = generateCapabilityToken(“04c5b070-4292-4b3f-911b-“,jamartinez@odins.es,”ngsi-
ld/v1/entities”,”GET”);
```

Configuration Parameters

The following configurations parameters are need for the library to access to the DEMETER Authorisation enabler:

Configuration parameter	Value	Type	Description
AUTHORISATION_URL	URL	String	Authorisation Endpoint

9.3 *Communications and Networking Enabler*

9.3.1 Communications and Networking Enabler: TLS/DTLS

9.3.1.1 Functionality description

This module provides confidentiality properties to a client-server communication, to prevent unauthorized readings or alterations by malicious users.

9.3.1.2 Interaction with other Enablers

This enabler will be integrated with the authentication enabler and, in general, with others security enablers. An authentication phase is mandatory to guarantee confidentiality aspects in a secure system, in fact these security components should be considered as a unique element in the system.

9.3.1.3 Dependencies on other Core/Advanced Enablers

This enabler only depends on the others security enablers suite, such as authentication enabler, authorization enabler, etc.

9.3.1.4 Deployment/Development considerations

The module will be implemented with OpenSSL, which is a well-known toolkit written in C that provides several libraries and APIs to perform some cryptographic tasks. OpenSSL supports several operating systems, e.g. Linux, Windows, OS X, iOS, Android, etc, with some different platforms, such as Intel, ARM, X32, etc.

9.3.1.5 Technical description

This module implements TLS/DTLS protocols providing confidentiality. Thanks to this, a HTTPS communication between client and server will be established. OpenSSL is a powerful and open source

solutions that provide an SSL/TLS toolkit and a cryptographic library. This toolkit implements all the features required by a secure communication over a computer network, in particular the module supplies that information is not made available to unauthorized entities, preserving them both from readings and from modifications.

The confidentiality is implemented through a secure communication channel and a session keys that make the information that is exchanged private. These security aspects are possible with algorithms that cypher the data in a proper manner, so that its reading is possible only for the entities which are in possession of the right keys.

Functions and Data model

Since the OpenSSL interface is a shell command line through which the user runs the commands for the machine, there are no functions or data model to describe.

UML Activity diagram(s)

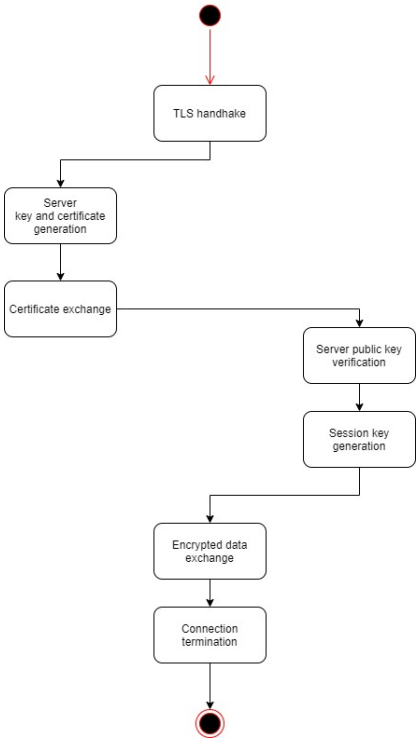


Figure 44: OpenSSL TLS/DTLS activity diagram

UML Sequence diagram(s)

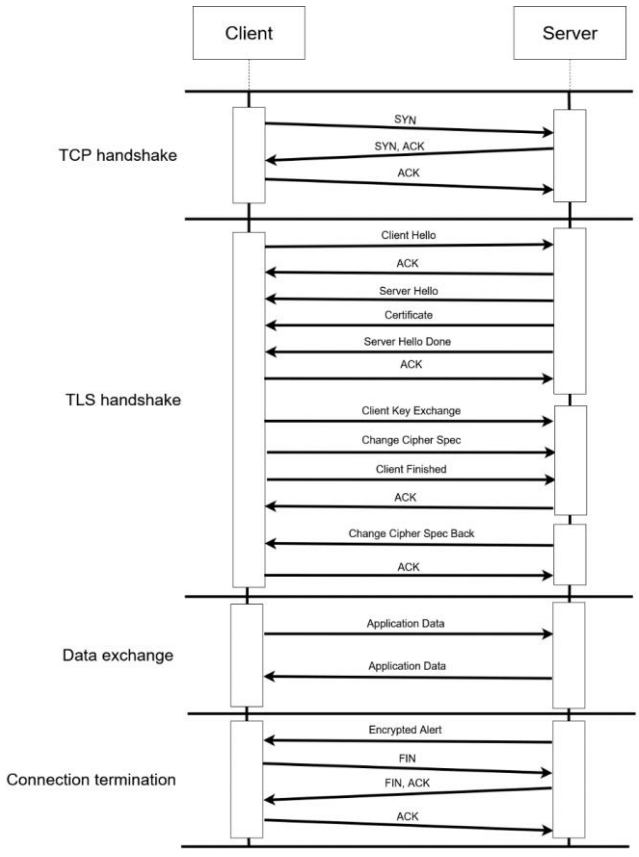


Figure 45: OpenSSL TLS/DTLS sequence diagram

Deployment

After downloading the OpenSSL master sources, to configure its library the toolkit uses a custom build system. Once configured, it is necessary to run a make command to build the library.

Configuration Parameters

This enabler does not have configuration parameters.

9.3.2 **Communications and Networking Enabler: JSON/XML Encryption**

9.3.2.1 Functionality description

Encryption and decryption of JSON and XML

9.3.2.2 Interaction with other Enablers

N/A

9.3.2.3 Dependencies on other Core/Advanced Enablers

N/A

9.3.2.4 Deployment/Development considerations

Python library dependent on the following libraries: objcrypt, json, pyDes

```
make clean
make all
```

9.3.2.5 Technical description/information

The functions in this library make use of existing external libraries to encrypt and decrypt JSON and XML objects.

Functions and Data model

Data models used by the functions/methods shall be described in tables:

Table 14: Encryption enabler, json data model

Name	JSON	
Property	**Type**	**Description**
N/A	JSON	JSON data model

Table 15: Encryption enabler, XML data model

Name	XML	
Property	**Type**	**Description**
N/A	XML	XML data model

Title	*Encrypt_json*
Function 1 *This field holds the name of the function used and the required (and optional) parameters*	
encrypt_json(json_to_encrypt, key, labels=None)	
Output *This field holds the type of the output expected*	
Encrypted JSON (str)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
json_to_encrypt	JSON dict to encrypt
Required:	
key	Alphanumeric string containing the password for the encryption
Optional:	
labels	Name of the labels separated by ";", string. If none, select all

Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
{"test": "X5rP1dfame+/5UIW35kmzoISBYOlZ4KjklL7qTTMBcSKA9lpCOZkD7lVgOWk1hWY"}	Encrypted JSON
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
TBD	
Sample call *This field holds a possible sample call to the described function*	

dictionary={
'test': 'test value'
}
encrypt_json(dictionary, "test")	

Notes *This field holds any additional helpful info related to the function described.*	

Title	*Decrypt_json*
Function 2 *This field holds the name of the function used and the required (and optional) parameters*	
decrypt_json(json_to_decrypt, key)	
Output *This field holds the type of the output expected*	
Decrypted JSON (dict)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
json_to_decrypt	JSON string to decrypt
Required:	
key	Alphanumeric string containing the password for the decryption
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
{'test': 'test value'}	Decrypted JSON
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
{'test': ''}	The password is wrong and the decryption failed
Sample call *This field holds a possible sample call to the described function*	
encrypted_json={"test": "X5rP1dfame+/5UIW35kmzoISBYOlZ4KjklL7qTTMBcSKA9lpCOZkD7lVgOWk1hWY"} encrypt_json(encrypted_json, "test")	
Notes *This field holds any additional helpful info related to the function described.*	

Title	*Encrypt_XML*
Function 3 *This field holds the name of the function used and the required (and optional) parameters*	
encrypt_xml(xml_to_encrypt, key, labels=None)	
Output *This field holds the type of the output expected*	
Encrypted XML (bytes)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
xml_to_encrypt	XML string to encrypt
Required:	
key	Alphanumeric 8 characters string containing the password for the encryption
Optional:	

labels	Name of the labels separated by ";", string. If none, select all
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
'\xfca8\x8f\xdc\xffO\n\xee\xaed\xbd\x89\xe7\xd7y\x19\xfe\xee\xa6\xa8\xb9PI\xacG-\xcd\x15ASn\xe4Yd\xaeZ#\x04G\xd2\xcb\x91|\xb4\x07\x94"z\xe5\n!\x94\xa3\x03N~Z\x19^\xa4a\xc7x\x95x\x91\xde\xc3e\'o\xb1L\xf1V\xfe\x1c\x19\xa5'	Encrypted XML
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
ValueError: Invalid DES key size. Key must be exactly 8 bytes long.	The password is too short or too long
Sample call *This field holds a possible sample call to the described function*	
data = ''' <?xml version="1.0"?> <test> <title>Sample text</title> </test> ''' encrypt_xml(data, "test1234")	
Notes *This field holds any additional helpful info related to the function described.*	

Title	*Decrypt_XML*
Function 4 *This field holds the name of the function used and the required (and optional) parameters*	
decrypt_xml(xml_to_decrypt, key)	
Output *This field holds the type of the output expected*	
Decrypted XML (bytes)	
Params *This field holds the parameters (if any). Separated based on the fields below into required and optional.*	
Required:	
xml_to_decrypt	XML bytes to decrypt
Required:	
key	Alphanumeric 8 characters string containing the password for the decryption
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their call-backs should expect>	
b'\n<?xml version="1.0"?>\n <test>\n <title>Sample text</title>\n </test> \n'	Decrypted XML
Error response *This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.*	
TBD	The password is wrong and the decryption failed
ValueError: Invalid DES key size. Key must be exactly 8 bytes long.	The password is too short or too long
Sample call *This field holds a possible sample call to the described function*	
data = b'\xfca8\x8f\xdc\xffO\n\xee\xaed\xbd\x89\xe7\xd7y\x19\xfe\xee\xa6\xa8\xb9PI\xacG-\xcd\x15ASn\xe4Yd\xaeZ#\x04G\xd2\xcb\x91|\xb4\x07\x94"z\xe5\n!\x94\xa3\x03N~Z\x19^\xa4a\xc7x\x95x\x91\xde\xc3e\'o\xb1L\xf1V\xfe\x1c\x19\xa5' decrypt_xml(data, "test1234")	
Notes *This field holds any additional helpful info related to the function described.*	

Use cases / Data flow

Technically describe use cases of the enabler and the data flow using formal UML activity and sequence diagrams.

UML Activity diagram(s)

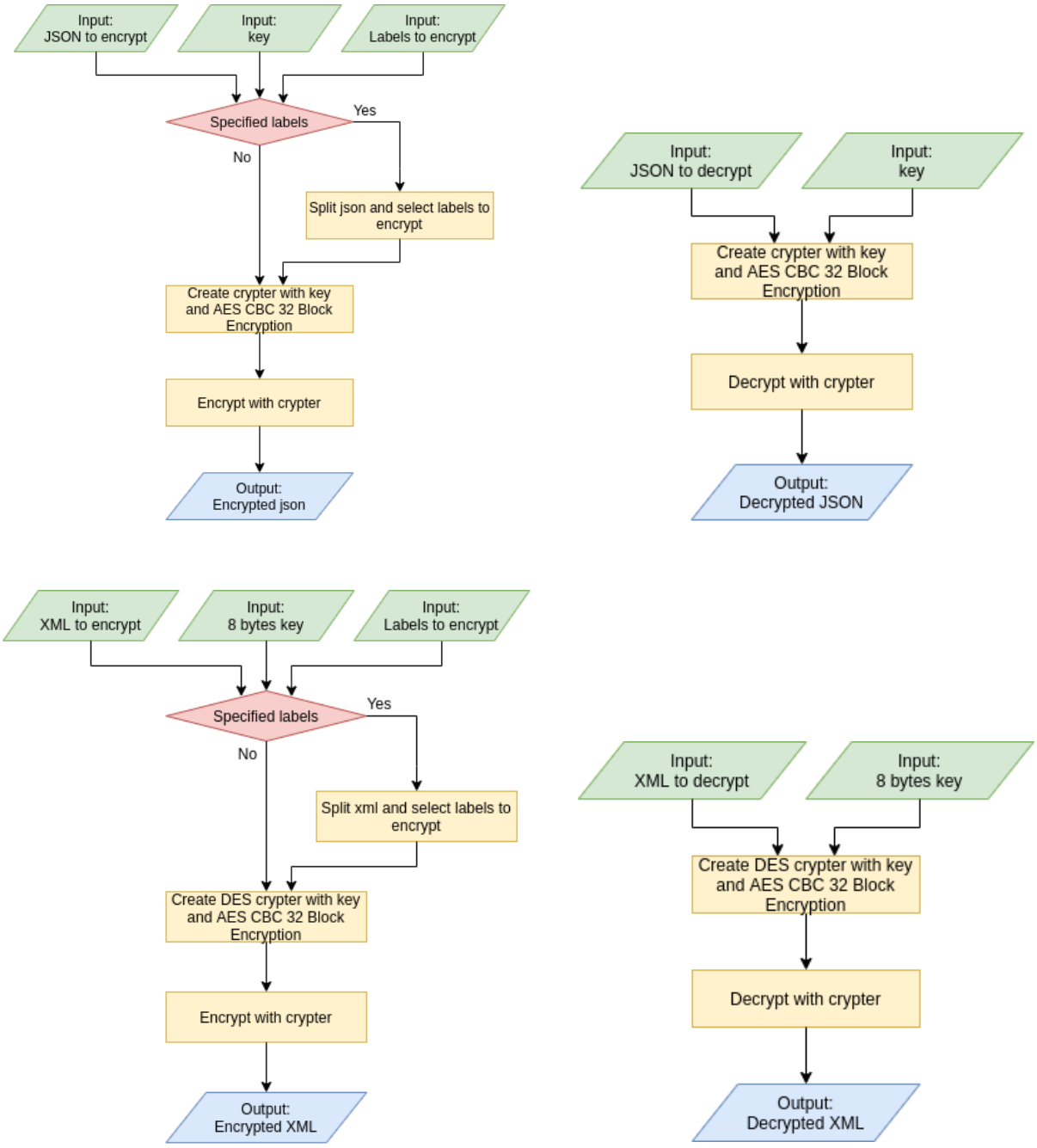


Figure 46: XML encryption and decryption activity diagram

UML Sequence diagram(s)

N/A

Deployment

```
gcc compiler:

make clean

make all
```

Configuration Parameters

This enabler does not have configuration parameters.

9.4 *DEH Client Enabler*

Some of the resources and data that will be exposed and used by the DEH Dashboard will be hosted by third parties or on remote or private infrastructures. The role of the DEH Client Enabler is to provide some libraries, SDKs or tools that can make exposing these resources as easy as possible for the providers. Initially the DEH Client Enabler will expose an API layer above the Resource Registration Modules and Resource Discovery Modules so that third parties can ensure that their resources can be registered and discovered. It will also interact with the Security Protection Enabler Components, then Resource Access Control Component and Semantic Interoperability API to ensure that access to the resources is strictly controlled and that data made available is done so in AIM format. This section outlines the functionality of this component, its interaction with other enablers, its dependencies on other enablers and finally technical considerations.

9.4.1 **Functionality description**

As this component is designed to be used outside of the central deployment of the core DEH, it will need to support a wide range of environments. In some cases it will need to provide a very lightweight abstraction above the APIs of the DEH Core enablers where there are limited resources in the remote sites, and in other cases it will provide local services that can meter and restrict usage of local resources being exposed to the DEH. The initial functionality focuses on providing client libraries that can be used to interact with the DEH.

The DEH Client Enabler will provide client libraries for the below APIs at a minimum:

- Resource Registration Management API
- Resource Discovery API
- Identity Management API
- Resource Access Control API

9.4.2 **Interaction with other Enablers**

Identity Manager will be used to verify if there is an authenticated user login session.

Resource Access Control will be used to on the one hand create access control policies on behalf of the resource owners, which will dictate under what conditions its resources can be accessed, and secondly, to ensure there is sufficient authorization for the APIs of the Resource Registration Management component to be used.

Resource Registration Management Component, a core DEH enabler, is a target service that the DEH Client Enabler will interact with. It will be used to perform CRUD operations on the user's resources in the central repository.

Discovery Management Component, a core DEH enabler, will be used to discover other resources that may need to be accessed and used by the user.

9.4.3 **Deployment considerations**

The DEH Client Enabler will be developed as client libraries for common software platforms. The initial client libraries will be developed in JavaScript and Python, which will enable for a broad set of application developers and third parties to adopt the technologies. Depending on future requirements, we will consider GOLAN, C/C++ and Java. The software modules will be made available through openly accessible software repositories including NPM and PyPI.

A set of sample client implementations will be developed to illustrate how the client libraries can be integrated by end users.

9.4.4 **Technical description**

9.4.5 *API and Data model*

Name	DEHClient	
Property	Type	Description
auth	DEHClientAuth object	Object to manage authentication and authorisation of the client
isLoggedIn	bool	Property indicating whether the client is logged in or not
rm	DEHRegistrationManagement object	Object to manage registration of resources
dm	DEHDiscovery object	Object to manage resource discovery

Name	DEHClientAuth	
Property	Type	Description
user	JSON	Parameters of the user object that is authenticated or NULL
sessionID	text	A valid session id for this logged in user or NULL
authToken	text	A valid authorisation token for this logged in user or NULL
Function	Parameters / Result	Description
checkAuthenticationToken	**Input**: User object **Returns**: session ID and authorisation token	Checks to see if there is a login token available for this session and for this user object. User object depends on the authentication method used. Can be username/password, or APIkey for example.

Name	DEHRegistrationManagement	
Property	Type	Description
connection	JSON	Parameters of the connection object established with the registry or NULL
Function	Parameters / Result	Description
registerResource	**Input**: JSON object with resource attributes **Returns**: resource ID or Error if not authorised, or malformed	Creates a resource registration from the description in the JSON object.
updateResource	**Input**: JSON object with resource attributes **Returns**: resource ID or Error if not authorised, or malformed	Updates the referenced resource with the JSON object
deleteResource	**Input**: JSON object with resource ID **Returns**: resource ID or Error if not authorised, or malformed	Deletes the referenced resource with the JSON object.
getResourceAccessControlPolicy	**Input**: JSON object with resource ID **Returns**: resource ID or Error if not authorised, or malformed	Retrieves the access control policies of the resource.
updateResourceAccessControlPolicy	**Input**: JSON object with resource ID **Returns**: resource ID or Error if not authorised, or malformed	Updates the access control policies of the resource

Name	DEHDiscovery	
Property	Type	Description
connection	JSON	Parameters of the connection object established with the registry or NULL
Function	Parameters / Result	Description
findByKeywords	**Input**: array of keywords or comma separated string **Returns**: array of resource ID or Error if not authorised, or malformed	Searches for a set of resources based on a key word search of the resources descriptions.
findByCriteria	**Input**: JSON object with a set of criteria equalities **Returns**: array of resource ID or Error if not authorised, or malformed	Searches for a set of resources based on a set of criteria, where different fields of the resource searched to see if they meet the criteria.

10 CI/CD Infrastructure and Tools

Continuous Integration (CI) is a developer practice to keep a working system by small changes growing the system by integrating frequently (usually at least daily) on the mainline by means of appropriate tools supporting automation with lots of automated tests. This enables teams to work on shared code and increases the visibility into the development and quality of the system. By referring to a developer practice Continuous Integration (CI) typically expects developers to implement Test-driven development (TDD) with constant refactoring practice. When a developer is unit-test-driving his code, he ensures that his local copy is always working.

Continuous Deployment (CD) refers to the automated deployment of new -release- versions of a system to the production environment. Following the continuous integration process, as described above, when a system reaches a maturity level (as indicated by specific, predefined criteria), the CD takes care of updating an existing running version of the system automatically, minimizing downtime.

Combined, CI/CD is a pipeline that gets new developments and provides an updated running version of a system hosted in a predefined environment.

10.1 *CI/CD tools in DEMETER*

In DEMETER, a private CI/CD environment has been setup and is already being used by the consortium. This environment comprises of several tools, which are described below.

10.1.1 Version control

GitLab has been selected to be used for managing source code and version control in DEMETER. GitLab is an open source code management system based on Git, which includes a user management part that can be hosted online. DEMETER's code repository is using GitLab's online version where several private repositories have been created following the structure indicated by the partners involved. The group functionality offered by GitLab allows for code isolation, hence, to better accommodate privacy and IPR concerns among the consortium, subgroups have been defined where access is only granted to partners directly involved to the related component and task. In cases where public repositories are required, e.g., for public components, according to the Description of Action (DoA) commitments. Source code that will be made public will of course be subject to licensing terms and conditions as agreed between the partners involved. Gitlab provides the ability to allow access to external parties.

In addition, Gitlab offers pipelines for automated integration and deployment processes. Pipelines describe sets of sequential continuous integration (CI) and continuous delivery (CD) operations. In this course, CI pipelines include code building followed by automated unit and integration tests while CD pipelines deploy the code to different environments, for reviewing purposes, for actual user testing (staging environment) and, finally for production use (production environment). The above is depicted in Figure 47.

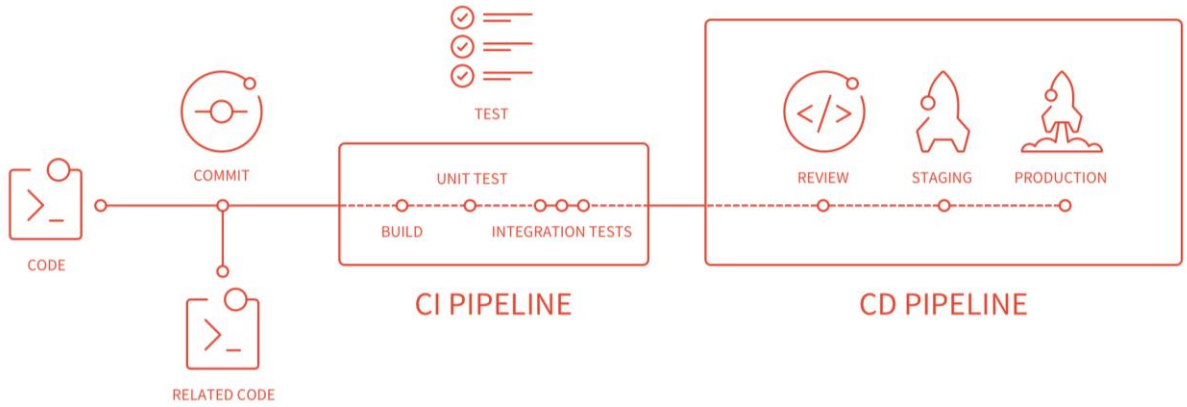


Figure 47: Continuous integration

10.1.2 **CI/CD pipelines**

As mentioned above, Gitlab's CI/CD framework uses pipelines to automate integration and deployment processes. Such a pipeline is depicted in Figure 47.

Pipelines are defined and described in script files (.gitlab-ci.yml files, an example available in Figure 48), each of them representing a "job", including various pipelines organized in stages. Each job is assigned to a Gitlab runner to be executed. Gitlab runners are merely (client) Gitlab services that run on private or public infrastructure, connect to a public or private Gitlab instance, and execute the jobs described in the job files (building, testing, deploying). Runners execute the jobs in Docker containers while they also run as Docker containers, hence, in DEMETER we are using a Docker-in-Docker paradigm for the Runners we use. This enables as to achieve higher utilization of our cloud infrastructure resources. Upon the execution of the jobs, GitLab offers a reporting tool to the developers to inspect all job stages.

Core functionalities of GitLab's CI/CD framework are listed below:

- Multiple projects are possible, grouped under groups and subgroups, allowing for organizing source code and components according to the architectural blocks they belong to or other criteria indicated by the partners.
- Private/public projects can be created, so components and source code can be publicly available if needed.
- GitLab provides branching, developing, testing, reviewing features to the development teams so that they can carry out their tasks in parallel before merging their work.
- Gitlab provides a private Image Registry where container images can be uploaded and used in e.g., Docker container deployments.

File templates .gitlab-ci.yml Choose a template...

```
image: docker:latest
services:
- docker:18-dind

stages:
- ver
- build
- test
- release

variables:
  TEST_IMAGE: registry.gitlab.com/demeterproject/test:$CI_COMMIT_REF_NAME
  RELEASE_IMAGE: registry.gitlab.com/demeterproject/test:latest

before_script:
  - docker login -u gitlab-ci-token -p $CI_JOB_TOKEN registry.gitlab.com

ver:
  stage: ver
  tags:
    - docker
  script:
    - cat /etc/os-release
    - whoami

build:
  stage: build
  tags:
    - docker
  script:
    - docker build --pull -t $TEST_IMAGE .
    - docker push $TEST_IMAGE

test:
  stage: test
  tags:
    - docker
  script:
    - docker pull $TEST_IMAGE
    - docker run $TEST_IMAGE echo 'Hello World'

release:
  stage: release
  tags:
    - docker
  script:
    - docker pull $TEST_IMAGE
    - docker tag $TEST_IMAGE $RELEASE_IMAGE
    - docker push $RELEASE_IMAGE
  only:
    - master
```

Figure 48: Example job file

10.2 *CI/CD infrastructure*

DEMETER's CI/CD infrastructure comprises of the online repository (including the Image Registry) hosted on Gitlab's cloud and DEMETER's private cloud which is meant to host Gitlab Runner containers for the automated processes in CI/CD but also for the deployment of any DEMETER components that will be deployed on DEMETER's cloud. This infrastructure is not meant for pilot-specific components as these components will be deployed on pilots' premises. The CI/CD infrastructure setup is depicted in Figure 49.

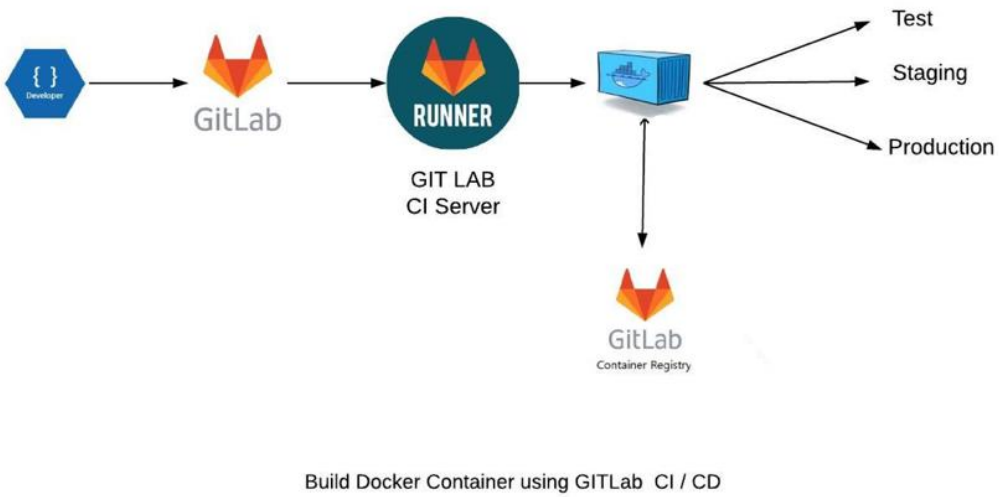


Figure 49: DEMETER CI/CD infrastructure

DEMETER's CI/CD infrastructure includes at the moment:

- 3 virtual machines (VMs) with 2vCPU, 4 GB RAM, 40 GB SSD
- 3 virtual machines (VMs) with 1vCPU, 2 GB RAM, 20GB SSD

All the VMs above are hosted on Hetzner Cloud, located in Germany.

Depending on partners' needs, further resources might be allocated. In such case, this will be reported in the next version of this deliverable (M22).

To avoid technical incompatibilities among components and to ensure isolation, thus, increasing component security, all components will be deployed as docker containers on DEMETER's cloud infrastructure.

11 Verification and Validation Plan

Verification and Validation plan aims to reassure that DEMETER's components are successfully integrated and perform as they were described during the design phase of the project. It describes the process that DEMETER components need to follow in order to be tested properly and subsequently the process that documents and validates their functional and non-functional performance in stand-alone manner and as a part of a greater system (pilot).

11.1 *Verification Plan*

The Verification Plan describes the process that DEMETER implementations have to follow in order to be able:

- to verify that each application offers the functionality that was envisioned provide during the design phase of each of the DEMETER platform's components, and
- to verify that the integration between each of the DEMETER platform's components has been successfully carried out.

This is realized through a set of Test Levels that can be executed upon them. Briefly, the purpose of these test is to verify that each component:

- Provides the required functionality that was designed for.
- Can Integrate successfully with the relevant DEMETER platform's components.
- forms a system that meets the required KPIs that were set.

Verification plan includes Test Levels, each of them addressing a different aspect of the verification process. These are described below.

11.1.1 Test Levels

According to the International Software Testing Qualifications Board's (ISTQB's) Agile Test Extension [1] the following test levels can be defined:

Component testing (also known as unit, module, or program testing) searches for defects in, and verifies the function of, software modules programs, objects, classes, etc., that are separately testable. It may be done in isolation from the rest of the system, depending on the context of the development life cycle and the system. In the context of DEMETER platform development, separate component tests will be planned and executed in each technical Work package delivering DEMETER components. Such tests will facilitate the verification at component level (unit-test).

Integration testing evaluates the interfaces between components, interactions with different parts of a system and interfaces between systems. Systematic integration strategies may be based on system architecture (such as top-down and bottom-up), functional tasks, transaction processing sequences or some other aspect of the system or components. To ease fault isolation and detect defects early, integration should normally be incremental rather than "big bang".

System testing is concerned with the behaviour of a whole product. In system testing, the test environment should correspond to the final target or production environment as much as possible to minimize the risk of environment-specific failures not being found in testing. System testing may include tests based on risks and/or on requirements specifications, business processes, use cases, or other high-level text descriptions or models of system behaviour, interactions with the operating

system, and system resources. In DEMETER, this level of testing should be scheduled prior to the pilot demonstrations and is expected to be facilitated by the DEMETER Pilot leaders.

Acceptance testing aims to establish confidence in the system, parts of the system or specific non-functional characteristics of the system. It is often the responsibility of the customers or users of a system; other stakeholders may be involved as well. Finding defects is not the main focus in acceptance testing. Acceptance testing may assess the system's readiness for deployment and use, although it is not necessarily the final level of testing. In DEMETER this level of testing is optional, since commercialization of the DEMETER platform is not expected within the project timeframe.

Figure 50: DEMETER's Component Test Levels

Figure 50 illustrates in a summative manner the Test Levels that each component has to be validated and verified upon in the context of DEMETER project.

11.2 *Validation Plan*

Storyboard based validation. The release-based validation will be performed by pilot owners. A specific release validation form, containing all the user requested functionalities per pilot, will be used to validate the DEMETER platform during the pilot demonstration phases. The validation procedure will produce a list of features not implemented, partially implemented, or fully implemented. The list of the incomplete features and an analysis of the issue(s) will be presented to the relevant DEMETER partners in order to assist them solving the problem(s).

Documentation should be done per feature/component validated. The documentation per feature/component will be collected and reviewed by the pilot owners, which will follow a documentation template. The missing documentation will be reported to the relevant DEMETER partners. The documentation provided to the pilot leaders will serve as a basis for the validation process. The relevant documentation description that is required is described in the Appendix D: Component Testing Report Documentation. After each pilot-based validation, the validated documentation will be incorporated into platform release package.

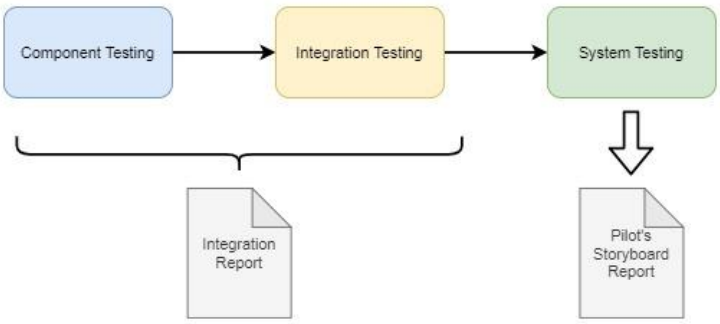


Figure 51: Verification and Validation process

Figure 51 above illustrates the Verification and Validation process that needs to be followed by each DEMETER's partner. Through Component and Integration Testing each partner compiles the Integration report that summarizes the testing results. Given on the component integration reports, Pilot leaders will compile the storyboard validation report that would verify that all the request functionalities, per pilot, have been realized.

Validate the Key Performance Indicators (KPI's). Each release will present a list of KPI's that will be validated using measurable characterization, such as: not reached, partially reached, fully reached. A KPI validation report will be shared with technical partners to assist them reaching the goal.

12 Conclusions

This document accompanies and describes the Release 1 of the DEMETER components and tools that enable solution integration, interoperability with external platforms and deployment support for pilot cases. These components and tools are released in a scheme that

- on one hand provide a **concrete** implementation to be used by the pilot applications and guide further development, and
- on the other hand, allows full **flexibility** for the application configuration and deployment to facilitate the highly different pilot needs and the various business models of the stakeholders.

It reflects work done in Tasks T3.2, T3.3, T3.4, T3.5 and T3.5 but it is based on work done in T3.1 (which produced D3.1 “DEMETER reference architecture - Release 1”) and utilizes work done in WP2 (D2.1 “DEMETER data models and semantic interoperability mechanisms” and D2.2 “DEMETER data and knowledge extraction tools an and D2.2”). It is accompanied by a set of other deliverables derived in WP4 (D4.1 “Decision Support, Benchmarking and Performance Indicator Monitoring Tools - Release 1” and D4.2 “Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space”). All together they provide the first release of the DEMETER reference implementation and contribute to project Milestone 2 “DEMETER Enablers, Hub, Spaces and Applications Release 1”.

This release is to be used in the coming months by the 20 DEMETER pilots to build and evaluate their DEMETER-enabled applications. Based on their feedback, the revised version of the DEMETER Reference Architecture will be presented in D3.3 (February 2021) and a revised version of this deliverable will be presented in D3.4 (June 2021).

13 Appendix A: Detailed Requirements

13.1 *Technical and Syntactic Interoperability of pilot technologies/platforms*

Requirement ID	TI1.1	Version	0.1	Last Update Date	27/01/2020
Title	Utilization of existing standards				
Description	DEMETER should utilize existing standards to provide interoperability. DEMETER should consider the following Internet Interoperability standards and adopt or build on top of the most appropriate/relevant: 1. ISO/IEC AWI 21823 2. ISO/IEC 29182 3. AIOTI WG03 4. FIWARE 5. FIWARE - NGSI 6. W3C 7. ETSI NGSI-LD 8. oneM2M				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 8: Unified agriculture ontology				
Reference component(s)	TBD[10]				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

[10] At the time of collecting the requirements the components and technologies to be used were not specified yet, hence they are marked as TBD for most requirements. The mapping to components is provided in Section **Error! Reference source not found.**.

Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

Requirement ID	TI1.2	Version	0.1	Last Update Date	27/01/2020
Title	Support of Communication Protocol Standards				
Description	DEMETER solutions should support existing communication protocol standards: 1. OASIS (ISO/IEC 20802) - MQTT 2. NB-IoT 3. LoRA 4. ISO 11783 ISOBUS 5. AEF: EFDI 6. ISO 18000 (RFID) 7. SigFox 8. Cellular (3G, 4G, etc) 9. BLE 10. Bluetooth 11. Wi-Fi 12. IEEE 802.15.4				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 7: Cost and power effective IoT data acquisition				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA, TECNALIA
Status	Proposed
Comments/Remarks	

Requirement ID	TI1.3	Version	0.1	Last Update Date	27/01/2020
Title	Support of Geospatial Interoperability Standards				
Description	DEMETER solutions should support existing Geospatial Interoperability standards: 1. OGC WFS 2. OGC WMS 3. OGC WCS 4. OGC WPS 5. OGC Agriculture 6. OGC SWE 7. OGC POI				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 4: Earth Observation data service Innovation 14: Smart fruit pesticides management				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

Requirement ID	TI1.4	Version	0.1	Last Update Date	27/01/2020
Title	Provide interoperability with existing cloud platforms				
Description	DEMETER solutions should provide interoperability with existing cloud platforms: 1. Azure 2. Proba-V 3. AVR Connect 4. Digital Ocean				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Proposed				

Comments/Remarks	

Requirement ID	TI1.5	Version	0.1	Last Update Date	27/01/2020
Title	HTTP REST API(s)				
Description	DEMETER should be able to connect to (external) platforms via REST API(s)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI1.6	Version	0.1	Last Update Date	27/01/2020
Title	Pub/sub and messaging queue mechanisms				
Description	DEMETER should be able to connect to (external) platforms via pub/sub and messaging queue mechanisms				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.4, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

Requirement ID	TI1.7	Version	0.1	Last Update Date	27/01/2020
Title	Compliance with system domain standards				
Description	DEMETER shall be designed in compliance with standards selected according to system domain, i.e., web standards, telecommunication standards, user interface standards, WCAG 2.1, etc.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	TECNALIA
Status	Proposed
Comments/Remarks	

Requirement ID	TI1.8	Version	0.1	Last Update Date	27/01/2020
Title	Data formats				
Description	DEMETER should offer APIs using common data formats such as JSON, JSON-LD, XML, or RDF				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	TECNALIA, INTRA				

Status	Proposed
Comments/Remarks	

13.2 ***Environment for service discovery and provisioning***

Requirement ID	TI2.1	Version	0.1	Last Update Date	27/01/2020
Title	Service description definition				
Description	DEMETER must propose a common service description definition to be used for registering and discovering services from different platforms, building on existing frameworks and standards (such as OASIS SOA-RM)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyze, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 8: Unified agriculture ontology				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	UMU, INTRA, PSNC				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI2.2	Version	0.1	Last Update Date	27/01/2020
Title	Services provisioning maintaining data security and privacy				
Description	Services provided by DEMETER must implement security and privacy mechanisms to protect the data exchanged with other entities (services, users)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyze, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 8: Unified agriculture ontology Innovation 9: Secure Agricultural data sharing services				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	UMU, PSNC				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI2.3	Version	0.1	Last Update Date	27/01/2020
Title	Services registration to DEMETER Enabler Hub				

Description	DEMETER-enabled services must be able to register to DEMETER Enabler Hub and make themselves discoverable to consumers, i.e., other services or end users.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control Innovation 5: Farm enabler dashboards
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers, Farmers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU, PSNC
Status	Proposed
Comments/Remarks	

Requirement ID	TI2.4	Version	0.1	Last Update Date	27/01/2020
Title	Services' categorization				
Description	DEMETER must provide a way to group services in categories (e.g., Farm monitoring, Supply chain monitoring, Milk quality, etc.)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.2, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU, PSNC
Status	Proposed
Comments/Remarks	

13.3 ***Networking and Communication***

Requirement ID	TI3.1	Version	0.2	Last Update Date	04/02/2020
Title	Secure transport layer (TLS, SSH, etc.)				
Description	The transport layer should be secure to protect communications				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 9: secure Agricultural data sharing services				
Reference component(s)	TBD				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	VICOM
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI3.2	Version	0.2	Last Update Date	04/02/2020
Title	GDPR technical requirements				
Description	DEMETER must comply with GDPR technical requirements				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers, farmers				

Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	VICOM
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI3.3	Version	0.2	Last Update Date	04/02/2020
Title	Combination of physical/wireless communications and Internet backbone networks				
Description	DEMETER should combine the use of physical/wireless communications and Internet backbone networks, in order to enable data sharing, network and device management, cloud computations and storage.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 3: device automation and control Innovation 5: traceability Innovation 7: IoT data acquisition Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection				
Reference component(s)	-				
Reference technology(ies)	-				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Desirable
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI3.4	Version	0.2	Last Update Date	04/02/2020
Title	Control devices sharing information				
Description	DEMETER should provide the means to control IoT devices sharing information				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 3: device automation and control Innovation 5: traceability Innovation 7: IoT data acquisition Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				

Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

13.4 ***Security***

Requirement ID	TI4.1	Version	0.2	Last Update Date	04/02/2020
Title	Attribute Based Access Control or Distributed Capabilities Access Control component				
Description	DEMETER should provide an Attribute Based Access Control or Distributed Capabilities Access Control component that can be integrated with trial site platforms and gateways				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	WIT
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI4.2	Version	0.2	Last Update Date	04/02/2020
Title	Authentication and authorization mechanisms for services, accessing resources and information audit tools				
Description	DEMETER must offer authentication and authorization mechanisms for using services and accessing resources as well as information audit tools				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI4.3	Version	0.2	Last Update Date	04/02/2020
Title	Data protection and privacy on software execution, network communications and integrated solution security				
Description	DEMETER will offer Data protection and privacy on software execution, network communications and integrated solution security				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 11: data integration Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				

Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Proposed + Review
Comments/Remarks	

Requirement ID	TI4.4	Version	0.2	Last Update Date	04/02/2020
Title	Identity management, access control and audit log				
Description	DEMETER must allow identity management, access control and audit log				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	UMU				
Status	Proposed + Review				

Comments/Remarks	

Requirement ID	TI4.5	Version	0.2	Last Update Date	04/02/2020
Title	Encrypted communications, integrity controls and electronic signature functionalities				
Description	DEMETER should offer encrypted communications, integrity controls and electronic signature functionalities				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.4, T3.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: secure interoperability Innovation 5: traceability Innovation 9: secure Agricultural data sharing services Innovation 20: product authentication/fraud detection				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	UMU				
Status	Proposed + Review				
Comments/Remarks					

13.5 ***Device/resource Management (including databases)***

Requirement ID	TI5.1	Version	0.1	Last Update Date	27/01/2020
Title	Data storage systems access management				
Description	DEMETER should provide the means to manage access (CRUD operations) to multiple types of data storage systems including semantic repositories, relational databases and NOSQL databases				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 7: Cost- and power-effective IoT data acquisition				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	TECNALIA, INTRA				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI5.2	Version	0.1	Last Update Date	27/01/2020
Title	Registration the capabilities of a resource				
Description	DEMETER should provide the means to register the capabilities of a resource (platform, thing, service) to the DEMETER Enabler Hub, thus				

	being available to interested parties. Therefore, it will be able to make use of other Enablers registered in the Hub to enhance the resource's features
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	TECNALIA
Status	Proposed
Comments/Remarks	

Requirement ID	TI5.3	Version	0.1	Last Update Date	27/01/2020
Title	Multiple devices bulk operations				
Description	DEMETER solutions should support multiple devices bulk operations (e.g., to be able to access a service offered by multiple devices)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.2, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	UMU
Status	Proposed
Comments/Remarks	

Requirement ID	TI5.4	Version	0.1	Last Update Date	27/01/2020
Title	Resource/device sharing rules				
Description	DEMETER solutions could specify rules (e.g., concurrent users, data limits, etc.) on the usage of shared resources/devices				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				

	Innovation 3: Agricultural automation and control Innovation 9: Secure Agricultural data sharing services
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

13.6 ***Runtime environment, Deployment management & Orchestration***

Requirement ID	TI6.1	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER Enablers deployment				
Description	DEMETER must make it possible for developers to deploy DEMETER Enablers on their premises				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI6.2	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER Enablers compliance				
Description	DEMETER should provide means to developers to verify that the enablers they implemented are DEMETER-compliant				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	INTRA				
Status	Proposed + Reviewed				
Comments/Remarks					

Requirement ID	TI6.3	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER deployment tests				
Description	DEMETER could provide tests to verify that the deployment of an enabler has been successful (e.g. having an endpoint at the enabler side that will be used for testing its connectivity)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	INTRA				
Status	Proposed + Reviewed				
Comments/Remarks					

Requirement ID	TI6.4	Version	0.1	Last Update Date	27/01/2020
Title	DEMETER runtime environment agnostic				
Description	DEMETER enablers should be runtime environment agnostic (i.e., developers can develop an enabler in any runtime environment (be it Linux or Windows or others SO) and achieve DEMETER-compliance. Technologies that enable virtualization and allow applications to run independently of the environment must be guaranteed. Enablers should be able to be deployed in various environments				

Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.3, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI6.5	Version	0.1	Last Update Date	27/01/2020
Title	Deployment process documentation				
Description	DEMETER should provide clear guidelines (e.g. reference documentation) for technology and solution providers in order to standardize the deployment process as much as possible in both development and production environments				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				

Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI6.6	Version	0.1	Last Update Date	27/01/2020
Title	Deployment software life-cycle management				
Description	DEMETER should provide an adequate technology solution and suitable tools able to manage the entire software life-cycle management (from development to production environment)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	ENG				

Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI6.7	Version	0.1	Last Update Date	27/01/2020
Title	Deployment process security				
Description	DEMETER should ensure a good level of security in the deployment process (e.g. the connections with DEMETER components for deployment purposes such as the AIS and the Enabler Hub should be secure)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 9: Secure Agricultural data sharing services				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ENG				
Status	Proposed + Reviewed				
Comments/Remarks					

13.7 ***Service / application life-cycle management***

Requirement ID	TI7.1	Version	0.1	Last Update Date	27/01/2020
Title	Service/application life-cycle management methodology				

Description	Each software component development in DEMETER should follow a service/application life-cycle management methodology (waterfall or agile)
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ATOS
Status	Proposed
Comments/Remarks	

Requirement ID	TI7.2	Version	0.1	Last Update Date	27/01/2020
Title	Technical requirements review				
Description	DEMETER technical requirements will be defined or reviewed for all the different software/services to be developed in the beginning of every iteration, and will be used for the development plan design as well as for the evaluation stages				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				

Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ATOS
Status	Proposed
Comments/Remarks	

Requirement ID	TI7.3	Version	0.1	Last Update Date	27/01/2020
Title	Components' testing				
Description	DEMETER components have to pass a set of tests to be defined at the beginning of the development phases in order to evaluate the results from those development phases.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ATOS
Status	Proposed
Comments/Remarks	

Requirement ID	TI7.4	Version	0.1	Last Update Date	27/01/2020
Title	Development teams' communication				
Description	DEMETER component development teams will have fluid communication (at any level) to guarantee the correct development and integration of the different components involved in the project.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				

Priority Level	Mandatory
Identified by Partner(s)	ATOS
Status	Proposed
Comments/Remarks	

Requirement ID	TI7.5	Version	0.1	Last Update Date	27/01/2020
Title	Component maintenance				
Description	DEMETER components will be maintained by their corresponding developers to guarantee their correct functioning during the project.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ATOS				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI7.6	Version	0.1	Last Update Date	27/01/2020

Title	Service/application life-cycle management software suites
Description	Service/application life-cycle management software suites will be used in DEMETER in order to ease the implementation of the life-cycle management methodology
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.3, T3.4, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ATOS
Status	Proposed
Comments/Remarks	

13.8 ***APIs and Application development support***

Requirement ID	TI8.1	Version	0.1	Last Update Date	27/01/2020
Title	CRUD to HTTP methods mapping				
Description	DEMETER's API(s) should handle CRUD operations by proper mapping to HTTP methods which indicate the type of action to be performed on the resources.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI8.2	Version	0.1	Last Update Date	27/01/2020
Title	Proper HTTP response codes				
Description	DEMETER services should always return the right status codes. HTTP status codes are standardized codes which have various explanations in various scenarios.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI8.3	Version	0.1	Last Update Date	27/01/2020
Title	Searching, sorting, filtering, and pagination				
Description	DEMETER API(s) should support searching, sorting, filtering and pagination. GET requests over collection resources can potentially return a large number of items. Web API should limit the amount of data returned by any single request.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DNET, SIVECO				
Status	Proposed + Reviewed				

Comments/Remarks	

Requirement ID	TI8.4	Version	0.1	Last Update Date	27/01/2020
Title	Stateless Authentication & Authorization				
Description	DEMETER API(s) should be stateless. Every request should be self-sufficient and must be fulfilled without knowledge of the prior request.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DNET, SIVECO				
Status	Proposed + Reviewed				
Comments/Remarks					

Requirement ID	TI8.5	Version	0.1	Last Update Date	27/01/2020
Title	Usage of Swagger for Documentation				
Description	DEMETER should use swagger for Documentation. Swagger is a widely used tool to document REST APIs that provides a way to explore the use of a specific API. The Open API Initiative was created by an industry consortium to standardize REST API descriptions across vendors. As part of this initiative, the Swagger				

	2.0 specification was renamed the OpenAPI Specification (OAS) and brought under the Open API Initiative.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Proposed + Reviewed
Comments/Remarks	

Requirement ID	TI8.6	Version	0.1	Last Update Date	27/01/2020
Title	REST-based services				
Description	DEMETER should be able to support REST based web services. DEMETER Enablers should be able to consume and provide REST APIs				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DNET, SIVECO
Status	Proposed
Comments/Remarks	

Requirement ID	TI8.7	Version	0.1	Last Update Date	27/01/2020
Title	Access control mechanisms in API(s)				
Description	DEMETER should require access control mechanisms for the API(s) and allow access to the offered endpoints only to authorized users/clients (e.g. other DEMETER Enablers)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

Requirement ID	TI8.8	Version	0.1	Last Update Date	27/01/2020
Title	API and application documentation				
Description	DEMETER should provide documentation to assist developers in API and application development. Documentation (tutorials, videos, guidelines, code examples) should be available to all developers that would like to e.g. create a new DEMETER enabler.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 3: Agricultural automation and control				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Proposed				
Comments/Remarks					

13.9 ***Enabler registration, discovery, provision, management, composition, accounting, billing***

Requirement ID	TI9.1	Version	0.2	Last Update Date	04/02/2020

Title	Semantic resource registry
Description	DEH should allow a provider to register offered services (or data) based on a registration description which DEH should provide. This could e.g. be in a triple-store to allow reasoning over unknown semantic entities and to be able to return offerings of coherent semantic entities to the consumer.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 8: Unified agriculture ontology
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Remark + Review
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.2	Version	0.2	Last Update Date	04/02/2020
Title	Discovery Management				
Description	DEH web application should allow a consumer (end-users) to discover suitable resources (e.g. components, devices, services, data sources, platforms, etc.), returning correct outputs matching with their requirements (search API or tags). The				

	discovery service should be based on a (semantic) query, offering a wizard to help the user build her query.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Discovery Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS, ENG, DEH Survey
Status	Proposed + Remark + Review
Comments/Remarks	Seems ok (1 comment and possible duplicate)

Requirement ID	TI9.3	Version	0.2	Last Update Date	04/02/2020
Title	Query Management				
Description	DEH should be able to save a query of a consumer; e.g. for future reuse.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	TBD
Reference component(s)	Reference component module (or sub-module) in the DEMETER Architecture
Reference technology(ies)	DEH – Resource Registry Management
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	ICCS
Status	Desirable + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.4	Version	0.2	Last Update Date	04/02/2020
Title	Rate services in publish & subscribe mechanism				
Description	DEH should allow a consumer to subscribe to an offered service as a result of a query. And then allow the consumer to rate the quality of the service it uses (subscribes to).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed +Remark
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.5	Version	0.2	Last Update Date	04/02/2020
Title	Resource Access Control				
Description	DEH should mandate that access to its APIs will be secured through user authentication and access control. Furthermore, for the subscription process DEH should make it possible to generate access credentials between consumers and producers in order to authenticate the communication between them. Finally, this process should be reasonably simple for developers to use.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Access Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				

Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Remark + Review
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.6	Version	0.2	Last Update Date	04/02/2020
Title	Query Management				
Description	DEH should periodically reissue a consumer's query and notify it of changes in the results (e.g. new offered services), if the consumer requests this service.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.7	Version	0.2	Last Update Date	04/02/2020

Title	Publish & Subscribe Notification
Description	DEH should notify a consumer if a subscribed service is changed by its provider (e.g. service withdrawn, or conditions changed).
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.8	Version	0.2	Last Update Date	04/02/2020
Title	Enablers Information Management				
Description	DEH should store information regarding the history of registration/provision and usage/consumption of enablers.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.9	Version	0.2	Last Update Date	04/02/2020
Title	DEH Scalability & Availability				
Description	DEH could be scalable, allowing the increase of users (providers and/or consumers) it accommodates without impacting performance or availability.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				

Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Review
Comments/Remarks	Seems ok (1 comment)

Requirement ID	TI9.10	Version	0.2	Last Update Date	04/02/2020
Title	Licensing				
Description	DEH should not be based on software that has IPR implications (or needs expensive licenses) thus blocking the provider/consumer ecosystem building.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ICCS				
Status	Proposed + Review				

Comments/Remarks	Seems ok

Requirement ID	TI9.11	Version	0.2	Last Update Date	04/02/2020
Title	Data encryption in communications				
Description	DEH should ensure encrypted communication between the user and the web server that exposes its interfaces (web GUI). Furthermore, it should ensure that internal communication between its software components (and therefore its APIs) also transmits the data in an encrypted manner, especially when it comes to user sensitive data.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS + ENG				
Status	Proposed + Review				
Comments/Remarks	(T3.4 to clarify) + (possible duplicate) + 1 comment				

Requirement ID	TI9.12	Version	0.2	Last Update Date	04/02/2020
Title	Service User Advisory				

Description	DEH could offer an "advisory" service in order to direct consumers towards contracting the appropriate services that they need.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.4, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	ICCS
Status	Proposed + Review
Comments/Remarks	(T3.4 to clarify)

Requirement ID	TI9.13	Version	0.2	Last Update Date	04/02/2020
Title	Accounting Management				
Description	DEMETER should provide accounting solution to allow users (consumers and producers) to create and send invoices to customers who purchase non-public resources available from the DEMETER Enabler Hub. An API framework could be provided for collecting accounting events, and also another API for users to interact with DEH				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.4, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH - Accounting
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Optional
Identified by Partner(s)	ICCS, ENG
Status	Proposed + Review
Comments/Remarks	(not to be implemented)

Requirement ID	TI9.14	Version	0.2	Last Update Date	04/02/2020
Title	Semantic Interoperability Framework				
Description	DEH must include (core) enablers for each device or external service that translates any data used by it to the Demeter AIM, when this data do not follow the AIM format natively; this ensure the necessary syntactic, semantic and technical interoperability of any Demeter-enabled applications.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				

Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Remark
Comments/Remarks	Seems ok

Requirement ID	TI9.15	Version	0.2	Last Update Date	04/02/2020
Title	Application portability				
Description	It would be desirable for the hub app to be portable to other platforms such as iOS and android.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	TBD				
Reference technology(ies)	iOS				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				

Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.16	Version	0.2	Last Update Date	04/02/2020
Title	System security services				
Description	Hub should consider the safety effects such as loss, damage or harm from an improper usage of the system, maintaining an expected integrity level. Also, there should be protection of the system from viruses, spyware, trojans and similar threats.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	ICCS				
Status	Proposed + Review				
Comments/Remarks	Check comment				

Requirement ID	TI9.17	Version	0.2	Last Update Date	04/02/2020
Title	System availability				

Description	DEH should guarantee response times for the app in the order of seconds, also considering the expected volume of request and use, even at peak times.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS, ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.18	Version	0.2	Last Update Date	04/02/2020
Title	TBD				
Description	Registration and provision management should be done from an external platform, IoT devices can be configured from external platform.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	TBD
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Non-Functional
Priority Level	Mandatory
Identified by Partner(s)	ICCS
Status	Proposed + Review
Comments/Remarks	Check comment

Requirement ID	TI9.19	Version	0.2	Last Update Date	04/02/2020
Title	Data synchronization				
Description	DEH needs to support different technological solutions to allow resources registration, coming from the DEMETER Provider (each platform, thing, service or application which can be available as a resource) through an API-based framework to offer an entry point to the Platform for the applications and services that intend share and synchronize data.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.20	Version	0.2	Last Update Date	04/02/2020
Title	Data federation				
Description	DEH needs to guarantee data federation techniques for API-based framework and technology tools to reduce data complexity.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ENG				

Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.21	Version	0.2	Last Update Date	04/02/2020
Title	Technology specification				
Description	DEH should define general and high-level specification on technological composition of the DEH Resource Registry and the User Registry, or the main features to be supported.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry, User Registry				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	ENG				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.22	Version	0.2	Last Update Date	04/02/2020
Title	DEH modules characteristic definition				
Description	DEH should define in detail all the functions or at least the high-level definition of the main features that each module must support. The				

	involved modules are: Compatibility checker, Resource registry management, Resource access control, User Account Management, Discovery Management.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.1, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, Compatibility Checker, Resource Access Control, User Account Control, Discovery Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI.23	Version	0.2	Last Update Date	04/02/2020
Title	Data management				
Description	DEH should offer means to have full control over all the services (such as compatibility checker, resource access controls, resource registry management, user account management), in				

	order to get, add, update and delete their information (or entities).
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.24	Version	0.2	Last Update Date	04/02/2020
Title	Data fusion				
Description	DEMETER needs to guarantee data fusion techniques for API-based framework and technology tools in order to produce a consistent data integration model (e.g. data coming from heterogeneous data sources).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	TBD
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.25	Version	0.2	Last Update Date	04/02/2020
Title	Monitoring & Audit				
Description	DEH management should offer means to monitor registered services and data sources workload.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.26	Version	0.2	Last Update Date	04/02/2020
Title	Information Management				
Description	DEH should enable users (acting as DEMETER Providers) to register their offered resources (components, devices, services, data sources, platforms, etc.), recording attributes such as name, description, maturity level, tags, etc.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	Reference technology for the module or sub-module				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				

Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.27	Version	0.2	Last Update Date	04/02/2020
Title	Data Semantic Interoperability				
Description	DEH should enable resources to be semantically described and escorted by meta-data, which include the security and data usage policies applicable (provided by WP2).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T2.1				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.28	Version	0.2	Last Update Date	04/02/2020
Title	Data Resource Definition				

Description	DEH should enable users to provide enablers either developed in the project or external ones (e.g. from third-party platforms).
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.29	Version	0.2	Last Update Date	04/02/2020
Title	Resource Management (CRUD operations)				
Description	DEH should enable users to add new resources anytime and edit them. It will be possible to see when the last edit related to the added resource was done.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.4, T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management, Resource Access Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.30	Version	0.2	Last Update Date	04/02/2020
Title	Web service interoperability				
Description	DEH should enable users to use web services or interoperability APIs (which use the HTTP protocol as data transport) to access the resources available to the DEH (USAGE API).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				

Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.31	Version	0.2	Last Update Date	04/02/2020
Title	Resource compatibility checker				
Description	DEH should enable users to integrate the available resources by allowing their compatibility checking (VALIDATION).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Compatibility Checker				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.32	Version	0.2	Last Update Date	04/02/2020
Title	Agriculture interoperability space resources				
Description	DEH should enable users to connect their resources as part of the AIS.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.33	Version	0.2	Last Update Date	04/02/2020

Title	Data Discovery Management
Description	DEH should enable users to browse the DEH and to discover suitable resources matching challenge requirements (SOCS).
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH – Discovery Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.34	Version	0.2	Last Update Date	04/02/2020
Title	Rating service				
Description	DEH web application should enable users to rate used components, services or enablers.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	TBD
Reference component(s)	DEH – Resource Registry Management
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Remark + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.35	Version	0.2	Last Update Date	04/02/2020
Title	Resource statistics report				
Description	DEH should enable users to view statistics on registered components (top used, most rated, recently added).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.36	Version	0.2	Last Update Date	04/02/2020
Title	Collection of enablers system				
Description	DEH should enable the design of a system (collection) of enablers and services to help users (or developers) who fuse together such enablers in order to provide a whole system which can then be sold to other users (e.g. farmers).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Non-Functional				
Priority Level	Optional				
Identified by Partner(s)	DEH Survey - proposed features				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.37	Version	0.2	Last Update Date	04/02/2020
Title	User profile management				
Description	DEH should offer suggestions tailored on user's profile.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	DEH Survey - proposed features				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.38	Version	0.2	Last Update Date	04/02/2020
Title	Responsive web GUI				
Description	DEH web application should be accessible via a web browser or smartphone/tablet, without requiring any client software installation.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	TBD
Reference component(s)	DEH
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok (possible duplicate)

Requirement ID	TI9.39	Version	0.2	Last Update Date	04/02/2020
Title	User account management				
Description	DEH web application should have a user registration/login section.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – User Account Control				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.40	Version	0.2	Last Update Date	04/02/2020
Title	User private home page				
Description	DEH web application should have a home page with description of the main functionalities.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Proposed + Review				

Comments/Remarks	Seems ok

Requirement ID	TI9.41	Version	0.2	Last Update Date	04/02/2020
Title	User registration web page				
Description	DEH web application should have a page to register new resources or edit the already registered ones. Registered users could add new resources that will be approved by an administrator.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.4, T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – User Account Control				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	DEH Survey				
Status	Proposed + Review				
Comments/Remarks	Seems ok				

Requirement ID	TI9.42	Version	0.2	Last Update Date	04/02/2020
Title	Resources Management web page				
Description	DEH web application should have a page for each resource.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5
Relevant Objective(s)	TBD
Relevant Innovation(s)	TBD
Reference component(s)	DEH – User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.43	Version	0.2	Last Update Date	04/02/2020
Title	Interoperability marketplace and catalogues solution				
Description	DEH web application should include the interaction with other initiatives which provide catalogues and marketplaces of solutions, as well as independent (INTEROPERABILITY).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				

Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.44	Version	0.2	Last Update Date	04/02/2020
Title	DEH solutions web page				
Description	DEH web application should have a page to register SOLUTIONS and associate to the solution a group of resources, preferable displaying the inter dependencies/relationships between them.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH – Resource Registry Management, Resource Access Control, User Account Control				
Reference technology(ies)	TBD				

Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	DEH Survey - proposed features
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	TI9.45	Version	0.2	Last Update Date	04/02/2020
Title	Team services				
Description	Localisation will be needed, access to open data sources, customisation for each industry/market sector. At a stretch, a team's feature, so that views can be shared between team members.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5				
Relevant Objective(s)	O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	TBD				
Reference component(s)	DEH				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Optional				
Priority Level	Mandatory				

Identified by Partner(s)	DEH Survey - proposed features
Status	Proposed + Review
Comments/Remarks	Seems ok

13.10 ***Stakeholder account management***

Requirement ID	TI10.1	Version	0.1	Last Update Date	27/01/2020
Title	Stakeholder access				
Description	DEMETER should define different roles with which various stakeholders will get access to the system.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.4, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	INTRA				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI10.2	Version	0.1	Last Update Date	27/01/2020
Title	Account management roles functionality				

Description	Different account management roles in DEMETER should correspond to different functionality
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.2, T3.5, T3.6
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

Requirement ID	TI10.3	Version	0.1	Last Update Date	27/01/2020
Title	Distinguishing a) internal and external stakeholders and b) primary and secondary stakeholders				
Description	DEMETER account management roles should distinguish between internal and external stakeholders, and between primary and secondary stakeholders				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				

Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space
Reference component(s)	TBD
Reference technology(ies)	TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

Requirement ID	TI10.4	Version	0.1	Last Update Date	27/01/2020
Title	Stakeholders' categorization				
Description	DEMETER account management should categorize the following stakeholders into different role groups: 1. User, 2. Developer, 3. Expert, 4. Administrator				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers				
Prerequisite(s)	None				

Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	INTRA
Status	Proposed
Comments/Remarks	

13.11 ***Monitoring, Awareness, Feedback***

Requirement ID	TI11.1	Version	0.1	Last Update Date	27/01/2020
Title	Feedback from end-users				
Description	DEMETER should provide solutions to gather feedback from farmers and end-users				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 2: Stakeholder Open Collaboration Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers, End-users, Farmers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Mandatory				
Identified by Partner(s)	TECNALIA				
Status	Proposed				
Comments/Remarks					

Requirement ID	TI11.2	Version	0.1	Last Update Date	27/01/2020
Title	Upvoting mechanism				
Description	DEMETER should provide a way for users to upvote a service (introduce a popularity indicator)				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.2, T3.5, T3.6				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms				
Relevant Innovation(s)	Innovation 1: Agriculture Interoperability Space Innovation 2: Stakeholder Open Collaboration Space				
Reference component(s)	TBD				
Reference technology(ies)	TBD				
Involved stakeholders/actors	Technology providers, Solution providers, End-users, Farmers				
Prerequisite(s)	None				
Type	Functional				
Priority Level	Optional				
Identified by Partner(s)	INTRA				
Status	Proposed				
Comments/Remarks					

13.12 ***General Non-functional requirements***

Requirement ID	GNFR.1	Version	0.2	Last Update Date	04/02/2020
Title	Business analytic data visualization suite				
Description	DEMETER needs to provide appropriate mechanism for the visualization of data coming from heterogeneous sources such as Big data systems, data warehouses, relational databases (or NoSQL) and web services that supply the data.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T4.3
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	GNFR.2	Version	0.2	Last Update Date	04/02/2020
Title	Decision Support System Dashboards				
Description	DEMETER needs to provide user interfaces that enable users to interactively explore and analyze digital data, allowing them to effectively identify interesting patterns, infer correlations and causalities, and supports sense-making activities.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				

Relevant Task(s)	T3.5, T3.6, T4.3, T4.5
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. AIS (Agriculture Interoperability Space) 4. BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	1. OPENNESS (OPEN Networked Enterprise Social Software suite) 2. KNOWAGE (Open Source Suite for any modern Business Analytics) 3. More TBD
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	GNFR.3	Version	0.2	Last Update Date	04/02/2020
Title	Web applications usability				
Description	DEMETER needs to ensure the usability feature of user interfaces to cover all aspects of the user's experience when interacting with the DEMETER data visualization tools and with its graphical interfaces or web GUI.				
Relevant Pilot(s)	ALL				

Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5, T3.6, T4.3, T4.5
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. AIS (Agriculture Interoperability Space) 4. BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	1. OPENNESS (OPEN Networked Enterprise Social Software suite) 2. KNOWAGE (Open Source Suite for any modern Business Analytics) 3. More TBD
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Remark
Comments/Remarks	Seems ok (1 comment)

Requirement ID	GNFR.4	Version	0.2	Last Update Date	04/02/2020
Title	Web application stylesheet				
Description	DEMETER needs to guarantee an appropriate Look & Feel for its user interfaces (e.g. web GUI) so they satisfy the user needs both in terms of visual appearance (look) and that of user interaction (feel).				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T3.5, T3.6, T4.3, T4.5				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models				

Relevant Innovation(s)	Innovation 5: Farm enabler dashboards
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. AIS (Agriculture Interoperability Space) 4. BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	1. OPENNESS (OPEN Networked Enterprise Social Software suite) 2. KNOWAGE (Open Source Suite for any modern Business Analytics) 3. More TBD
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Remark
Comments/Remarks	Seems ok (1 comment)

Requirement ID	GNFR.5	Version	0.2	Last Update Date	04/02/2020
Title	Web application friendliness				
Description	DEMETER needs to guarantee the friendliness of user interfaces (e.g. web GUI) in order to ease the use of the provided features or visualizations. In order to cover this feature, the DEMETER user interfaces should satisfy the following criteria: a. Be intuitive, i.e. graphical interfaces should be well designed, b. Easy-to-navigate c. Easy to update and remove d. Should not require third-party installation software, e. Manage errors effectively (simply, the user should always be notified of the software malfunction through targeted alerts that show the user what is going on and indicate a possible solution to solve it).				

Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5, T3.6, T4.3, T4.5
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards
Reference component(s)	1. SOCS (Stakeholder Open Collaboration Space Implementation) 2. DEH (DEMETER Hub) 3. AIS (Agriculture Interoperability Space) 4. BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	1. OPENNESS (OPEN Networked Enterprise Social Software suite) 2. KNOWAGE (Open Source Suite for any modern Business Analytics) 3. More TBD
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Remark
Comments/Remarks	Seems ok (1 comment)

Requirement ID	GNFR.6	Version	0.2	Last Update Date	04/02/2020
Title	Business analytic data visualization suite				
Description	DEMETER needs to guarantee valid approach in exploiting Big Data sources. This approach will need to support users browsing, understanding and discovering data insights. Nonetheless, Big data characteristics, such as volume, variety and velocity pose a number of challenges for visualization. Indeed, current visualization and exploration systems should effectively and efficiently handle the following aspects:				

	a. Real-time Interaction. Efficient and scalable techniques should support the interaction with datasets, while maintaining the system response in the range of a few seconds, b. On-the-fly Processing. Support of on-the-fly visualizations over dynamic sets of volatile raw (i.e., not preprocessed) data, c. Visual Scalability. Provision of effective data abstraction mechanisms is necessary for addressing problems related to visual information overloading, d. User Assistance and Personalization. Encouraging user comprehension and offering customization capabilities to different user-defined exploration scenarios and preferences according to the analysis needs are important features.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T3.5, T3.6, T4.3
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	1. DEH (DEMETER Hub) 2. AIS (Agriculture Interoperability Space) 3. BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	1. KNOWAGE (Open Source Suite for any modern Business Analytics) 2. More TBD
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Desirable
Identified by Partner(s)	ENG, ICE

Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	GNFR.7	Version	0.2	Last Update Date	04/02/2020
Title	DSS dashboard outcomes data visualization				
Description	DEMETER needs to support the visualization needs in the outcomes of some DEMETER tasks such as data analytics (WP2), decision making services (T4.1), benchmarking techniques (T4.2) and workflows of enablers (T4.4): a. Regarding data analytics, the analytic services need visualization support to provide better understanding of data structures and meaningful insight i.e. the outcome of the data analytic services b. Regarding benchmarking techniques, DEMETER could benefit from a user dashboard (Web GUI) that allow companies to consult and compare the outcome of DEMTER DSS with other DSS and also provide guidance on the choice of different technologies c. Regarding decision making, DEMETER decision support services would need appropriate visualization support to present decision support to the users. The visualization can present decision support in the form of alerts, reports, comparisons, performance KPIs, historic analysis etc. d. Regarding workflows of enablers, the decision support enablers will require visualisation to manage the data flows as well as reporting the outcomes of the actual processing taking place within certain enablers.				
Relevant Pilot(s)	ALL				
Relevant Use Case(s)	ALL				
Relevant Task(s)	T4.1, T4.2, T4.3, T4.4				
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders				
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems				

Reference component(s)	BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG, ICE
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	GNFR.8	Version	0.2	Last Update Date	04/02/2020
Title	DSS dashboard notification				
Description	DEMETER needs to guarantee a good standard in data visualization in a substantial number of Decision Support System (or optimize the existing DSS) in order to give suitable advice, recommendations and automated actions to end-users (e.g. farmers, dairy companies). This will allow the visualizations to be used by other pilots and to increase the interoperability between the Pilots through the use of common visualizations that address their potential needs e.g.: a. DSS for cost optimization (e.g. linking economical aspects of wholesale and retail prices) b. DSS for production preferences (e.g. the use of non-chemical pesticides, attention to animal welfare and tracking, transparency to the consumers) c. DSS for cost/benefit analysis of field operations (irrigation/fertilization) d. DSS for optimization on irrigation/fertilization strategies, disease monitoring, yield analysis (e.g. the estimation of crop yield according to climate conditions) e. DSS for the forecasting of phytopathologies on crops f. DSS for animal welfare tracking g. DSS to support the farmers for live support of agricultural processes.				

Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T4.1, T4.2, T4.3, T4.4
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models O2: Build knowledge exchange mechanisms O3: Empower the farmer, as a prosumer O6: Ease and streamline mechanisms for all stakeholders
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards Innovation 6: Performance evaluation of Decision Support Systems
Reference component(s)	BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers, Farmer, Advisors, Researchers, Interest groups
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG, ICE
Status	Proposed + Review
Comments/Remarks	Seems ok

Requirement ID	GNFR.9	Version	0.2	Last Update Date	04/02/2020
Title	DSS Dashboard widget				
Description	1. DEMETER needs to guarantee a data visualization tool with the following features in terms of graphical widgets for displaying data (at least): b. Text c. Image d. Chart e. HTML				

	f. Table g. DSS for animal welfare tracking h. DSS to support the farmers for live support of agricultural processes.
Relevant Pilot(s)	ALL
Relevant Use Case(s)	ALL
Relevant Task(s)	T4.3
Relevant Objective(s)	O1: Analyse, adopt, enhance existing information models
Relevant Innovation(s)	Innovation 5: Farm enabler dashboards
Reference component(s)	BID (Business Intelligence Dashboard Tool)
Reference technology(ies)	KNOWAGE (Open Source Suite for any modern Business Analytics)
Involved stakeholders/actors	Technology providers, Solution providers
Prerequisite(s)	None
Type	Functional
Priority Level	Mandatory
Identified by Partner(s)	ENG
Status	Proposed + Review
Comments/Remarks	Seems ok

14 Appendix B: DEMETER Enabler Template

14.1 *Text information – metadata*

We need this information as metadata, for BSE/DEH descriptions or other components

14.1.1 Functionality description

Describe the functionality of the enabler

14.1.2 Interaction with other Enablers

Describe how the Enablers functionality is combined with other Enablers. E.g. How will the Security Enabler be utilized by other Enablers? Will it be accessible by all Enablers or just by the Communication/Networking enabler for example?

14.1.3 Dependencies on other Core/Advanced Enablers

Describe any dependencies on other Core/Advanced Enablers. Which Enablers' APIs are implemented by this Enabler?

14.1.4 Deployment considerations

Describe consideration related to deployment, e.g., where the image will reside, how access will be provided, resources required, etc.

14.2 *Technical description*

This information formally describes features/characteristics of an Enabler

14.2.1 API and Data model

Describe the API and the Data model of the enabler in a technical way. E.g., a list of endpoints and their description using Swagger documentation, a list of topics to access in case of MQTT, NGSI-LD payload, etc.

Data models used by the APIs shall be described in tables:

Name	This field holds the name of the data model that is described in this table	
Property	Type	Description

Developers are strongly advised to adopt Swagger for online documentation of the developed APIs. If Swagger (or any other online documentation tool) is being adopted, additionally, provide here Swagger details for the online documentation

(REST API)

Title	This field holds the description of the API
URL: This field holds the relative path to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	

Method This field holds the type of the Method used	
GET | POST | DELETE | PUT	
URL Params This field holds the parameters (if any). Separated based on the fields below into required and optional.	
Required:	
id=[integer]	parameter description
Optional:	
image_id=[alphanumeric]	parameter description
Data Params This field holds the body payload of a post request.	
Required:	
id=[integer]	parameter description
Optional:	
image_id=[alphanumeric]	parameter description
Success response <What should the status code be on success and is there any returned data? This is useful when people need to know what their callbacks should expect>	
200 Content: { }	response description
Error response This field holds the list of all possible error responses. Doing that, helps prevent assumptions of why the endpoint fails and saves a lot of time during the integration process.	
404	response description
Sample call This field holds a possible sample call to the described endpoint in a curl-like format. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
Notes This field holds any additional helpful info related to this endpoint.	

(publish/subscribe API)

Title	This field holds the description of the API
URL: This field holds the relative URL to the described API. For simplicity Root path can be cut off from this description and can be placed as a hypertext above the API template	
Topic	This field holds the topic identifier (uid/path/name)
Payload request/response	This field holds the format type payload of the message (e.g JSON, NGSI-LD)
Payload representation request/response: This field holds the structure of the payload used	

{
"id": {string},
"type": {object},
"name": {string},
"value": {string},
"…" : "…"
}	

Payload Property description	Please write here the Data Model table that describe the payload properties
Required parameters (request)	This field holds the required parameters
Required parameters (response)	This field holds the required parameters
Success response This field holds the list of all possible successful responses	
Error response This field holds the list of all possible error responses	
Sample call This field holds a possible sample call to the described topic. Please, choose the format wisely so that is clear and easy to read by the interested parties.	
Notes This field holds any additional helpful info related to this endpoint.	

14.2.2 **Use cases / Data flow**

Technically describe use cases of the enabler and the data flow using formal UML activity and sequence diagrams

14.2.3 **UML Activity diagram(s)**

Place activity diagram(s) here

14.2.3.1 UML Sequence diagram(s)

Place sequence diagram(s) here

14.2.3.2 UML Component diagram(s)

14.2.4 **Deployment**

Technically describe the deployment process for the enabler using a Docker-compose script and the deployment execution commands.

```
version: '3'
services:
  db:
    image: mysql
    container_name: mysql_db
    restart: always
    environment:
      - MYSQL_ROOT_PASSWORD="secret"
  web:
   image: apache
   build: ./webapp
   depends_on:
     - db
   container_name: apache_web
   restart: always
   ports:
    - "8080:80"
```

Deploy by:

```
sudo docker-compose up --build -d
```

14.2.5 **Configuration Parameters**

Describe all configuration parameters that can be provided by a user/developer (mandatory/optional). These could be defined as env vars in the docker-compose script provided above. Examples could be external component URLs, IPs, ports, SSL params

Configuration parameter	Value	Type	Description
MYSQL_ROOT_PASSWORD	secret	String	Your MySQL password

15 Appendix C: DEH Survey

In order to have a shared design of the DEH, a survey was prepared by ENGINEERING with the support of Fraunhofer (WP7 "Multi-Actor Ecosystem Development" leader) and T3.5 participants and then reviewed by WP leaders and cluster pilot leaders.

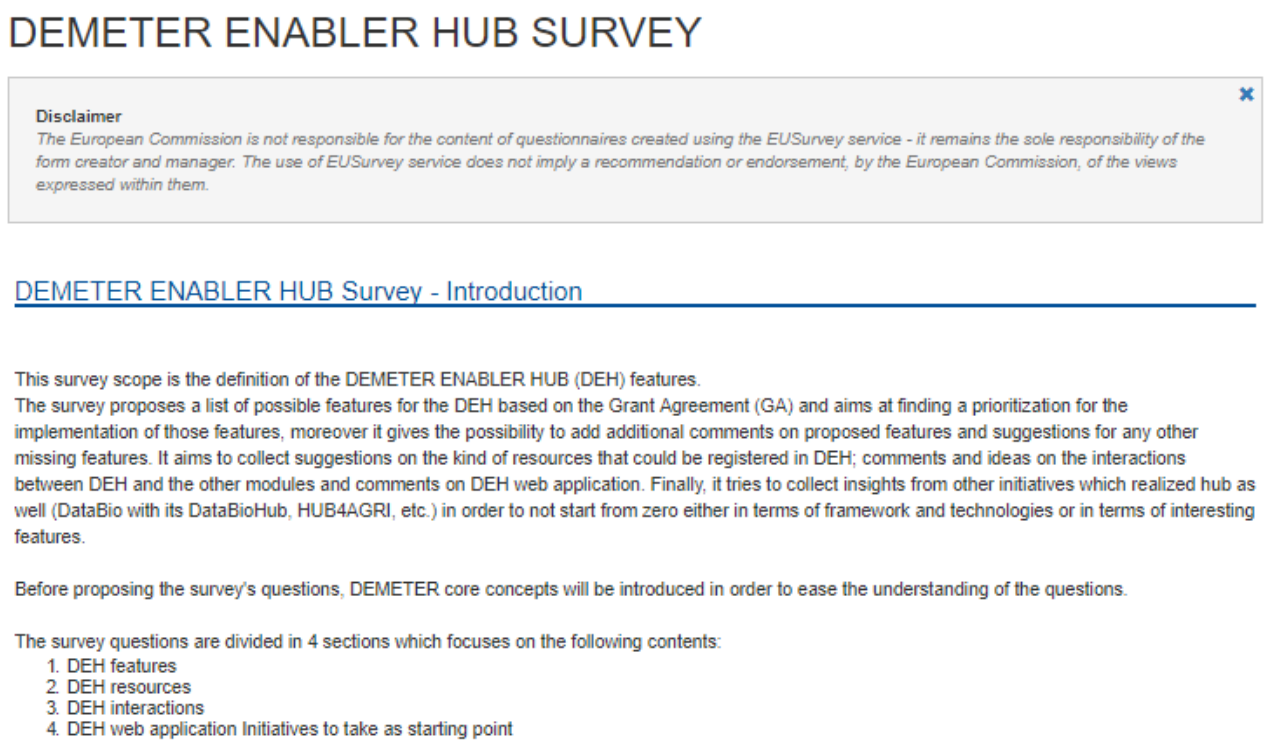
The survey proposed a list of possible features for the DEMETER ENABLER HUB (DEH) based on the Grant Agreement (GA) and aimed at finding a prioritization for the implementation of those features, moreover it gave the possibility to add additional comments on proposed features and suggestions for any other missing features. It aimed at collecting suggestions on the kind of resources that could be registered in DEH; comments and ideas on the interactions between DEH and the other modules and comments on DEH web application. Finally, it tried to collect insights from other initiatives which realized hub as well (DataBio with its DataBioHub, HUB4AGRI, etc.) in order to not start from zero either in terms of framework and technologies or in terms of interesting features.

Part of this survey appear in DEMETER deliverable D4.2 as it relates to that as well.

15.1 ***Survey structure***

The survey was structured in the following six sections:

1. Introduction section with an explanation of the main DEMETER core concepts.



DEMETER ENABLER HUB SURVEY

Disclaimer

The European Commission is not responsible for the content of questionnaires created using the EUSurvey service - it remains the sole responsibility of the form creator and manager. The use of EUSurvey service does not imply a recommendation or endorsement, by the European Commission, of the views expressed within them.

DEMETER ENABLER HUB Survey - Introduction

This survey scope is the definition of the DEMETER ENABLER HUB (DEH) features.
The survey proposes a list of possible features for the DEH based on the Grant Agreement (GA) and aims at finding a prioritization for the implementation of those features, moreover it gives the possibility to add additional comments on proposed features and suggestions for any other missing features. It aims to collect suggestions on the kind of resources that could be registered in DEH; comments and ideas on the interactions between DEH and the other modules and comments on DEH web application. Finally, it tries to collect insights from other initiatives which realized hub as well (DataBio with its DataBioHub, HUB4AGRI, etc.) in order to not start from zero either in terms of framework and technologies or in terms of interesting features.

Before proposing the survey's questions, DEMETER core concepts will be introduced in order to ease the understanding of the questions.

The survey questions are divided in 4 sections which focuses on the following contents:

1. DEH features
2. DEH resources
3. DEH interactions
4. DEH web application Initiatives to take as starting point

DEMETER core concepts

In order to allow a better understanding of the survey, it can be helpful to summarize DEMETER core concepts.
DEMETER follows an "**interactive innovation model**" as developed by the EIP-AGRI and fosters the development of research and the uptake of innovations into operational applications and the creation of new ideas thanks to interactions between actors ("cross-fertilisation"), sharing knowledge, expertise, capabilities and a wide range of "resources" (components, devices, services, data sources, platforms, etc.). The interactive innovation model is implemented through the "**multi-actor-approach**" MAA.
DEMETER implements the MAA approach across the full chain, from farmers to service advisors and suppliers. In this MAA, suppliers cover the full diversity of providers including ICT, data sources, knowledge, developers, software and hardware providers.
The MAA is implemented through a complete set of mechanisms structuring the human interaction with all stakeholders and supports this interaction through digital spaces, with each space catering to different phases of the interaction leading to the co-creation and deployment of new solutions at the farmers. Therefore, DEMETER delivers three spaces, the Stakeholer Open Collaboration Space (SOCS) and the Agricultural Interoperability Space (AIS) connected by the DEMETER ENABLER HUB (DEH).
The **DEH is a registry where suppliers can describe their resources** which become accessible by developers to be used through the "co-creation process" in order to build new solution.
Through the **SOCS**, DEMETER enables the co-creation approach to its wide network through four phases structured into:

1. Express and understand the need: this phase focuses on understanding the farmer's needs. This need can be expressed by the farmer directly in the DEMETER SOCS or through an agricultural service advisor who publishes it in the SOCS. DEMETER provides a structured approach for expressing farmers' needs, enabling farmers and advisors to consult other similar needs (and related solutions already implemented).
2. Select the optimal solution: Each need expressed by a farmer is analysed. A need or more needs could be translated in a challenge as result of pooling process. The creation of a challenge aims to involve DEMETER stakeholders in the creation of solutions (ideas) that represent their contributions to the requests expressed through the challenge. The end result of the challenge is the elaboration of the optimal solution, relying on the resources present in the DEH. This ensures that the combination is ready to use the interoperability mechanisms provided in the AIS to reach final deployment.
3. Develop the solution: once the optimal solution is selected, the group of suppliers involved in this selection team up to ensure its delivery. This phase therefore focuses on the integration phase, exploiting the interoperability mechanisms of DEMETER AIS.
4. Deploy the solution at the farmer(s) / cooperative: the final step of the DEMETER cycle is to deploy the solution at the farmer, connecting the extensions to the operational context of the farmer, ensuring its uptake, training if and as needed etc.

DEMETER AIS based on a reference architecture and a reference implementation, enables the secure interoperability and integration of different agriculture platforms, technology components and datasets, utilizing a brokerage environment based on producers/consumers. It involves technologies and data from different vendors and sources, ensuring their interoperability, and using (and enhancing) a core set of standards coupled with carefully-planned security and privacy protection mechanisms (also addressing business confidentiality). AIS focuses on delivering a full set of interoperability mechanisms to actually deploy the solution. DEMETER does not define completely new interoperability mechanisms, but uses (and extends) a wide range of pre-existing mechanisms at sensor, data and service levels.
The efficacy of the digital solutions realized, in reducing the costs, increasing the production and improving the environmental sustainable, will be then evaluated by means of the **Benchmarking System**. This will be possible if a farmer connects his Farm Management Information Systems with DEMETER. The benchmark system will check the availability data and calculate a set of indicators (yield by crop, vegetation index estimation, water, nutrient and pesticide consumptions, etc.), these farm indicator values will be compared with a set of target values(i.e. average and optimal indicator values from similar/neighbour farmers), finally the farmer receive a short report of the evaluated indicators showing which practices and digital solutions can improve the farm's performance.

2. DEH features: identification of which the DEH features should be, by selecting one of the options (“essential”, “desirable” or “unnecessary”) for each of the following propositions.

DEMETER ENABLER HUB Features

Please select the option that best describes your opinion to the statement "DEMETER ENABLER HUB should enable..."

DEH should enable:

	Essential	Desirable	Unnecessary
... users (acting as DEMETER Providers) to register their offered resources (components, devices, services, data sources, platforms, etc.), recording attributes such as name, description, maturity level, tags, etc.	◉	○	○
... resources to be semantically described and escorted by meta-data, which include the security and data usage policies applicable (provided by WP2)	○	○	○
... users to browse the Hub to discover suitable resources matching their requirements (search API or tags)	○	○	○
...users to provide enablers either developed in the project or external ones (e.g. from third-party platforms)	○	○	○
...users to add new resources anytime and edit them. It will be possible to see when the last edit related to the added resource was done.	○	○	○
...users to use web services or interoperability APIs (which use the HTTP protocol as data transport) to access the resources available to the DEH (USAGE API)	○	○	○
...users to integrate the available resources by allowing their compatibility checking (VALIDATION)	○	○	○
... users to connect their resources as part of the AIS.	○	○	○
...users to browse the DEH and to discover suitable resources matching challenge requirements (SOCS)	○	○	○
...users to expand their business thanks to the available payment service.	○	○	○
...users to rate used components	○	○	○
...users to view statistics on registered components (top used, most rated, recently added)	○	○	○
...users to see dependencies between various types of resources	○	○	○

Please write any additional comments about DEH features:

Please write any other ideas you may have about DEH features that could be essential or desirable:

DEH features:

3. DEH Resources: identification of the likely resources that could be registered in the DEH.

DEMETER ENABLER HUB Resources

Resources are all the platforms, components, devices, services, datasets, available in DEH. These flow into the DEH as digital resources. Each resource will have associated metadata, a description and will be categorized.

Please write any other resource you believe it should be considered as DEH resources that could be essential or desirable:

DEH resources:

Please write any additional comments about DEH resources:

4. DEH interaction: collection of the main ideas about the interactions between DEH and the other DEMETER modules.

DEMETER ENABLER HUB Interactions

DEH will interacts with the following modules:

- **Application Logic and user interfaces**: this module will allow web applications (external to DEMETER) or users to interact with the DEH in two ways: through APIs, DEH will interact with other web platforms and/or applications permitting these applications to create/modify resources within the DEH; through Web interfaces (GUI), users will have the possibility to access the DEH via web interface and search resources and solutions, using a web browser.
- **Stakeholders Open Collaboration Space (SOCS)** is a space where all stakeholders (farmers, advisors and suppliers) can collaborate, share best practices and participate to co-creation processes. This collaboration space makes a farmer need visible to advisors and developers and conveys the information coming from the farmers as input in order to select the most suitable resources registered in DEH to be used to build the optimal solution (idea for a challenge).
- **Agricultural Interoperability Space (AIS)** is a space dedicated to developers since its focus is on delivering a full set of interoperability mechanisms to actually deploy the solution (which consists of resources such as components, devices, services, data sources, platforms, etc that are accessible for deployment) through the DEMETER enabled interoperability mechanisms.
- **Benchmarking system** is a system to evaluate the productivity and the sustainability of the technologies/solutions adopted. The benchmarking system will be used to collect data from the Demeter providers to calculate a set of agronomic, economic and environmental indicators. The indicators will be used to compare the Demeter solutions (e.g. comparing the performance before and after the adoption of a specific solution) and can be used to provide the farmers with an indication on how they can improve the production or reduce the costs adopting them.

Please write any additional comments about already defined DEH interactions:

Please write any other ideas you may have about DEH interactions that could be essential or desirable:

5. DEH web application: identification of the main features about DEH web application.

DEMETER ENABLER HUB web application

Please select the option that best describes your opinion to the statement "DEMETER ENABLER HUB web application should":

DEH web application should:

	Essential	Desirable	Unnecessary
... be accessible via a web browser or smartphone/tablet, without requiring any client software installation	○	○	○
... have a user registration/login section	○	○	○
... have a home page with description of the main functionalities	○	○	○
... have a resource search functionality	○	○	○
... have a page to register new resources or edit the already registered ones. Registered users could add new resources that will be approved by an administrator.	○	○	○
... have a page for each resource	○	○	○
... include the interaction with other initiatives which provide catalogues and marketplaces of solutions, as well as independent (INTEROPERABILITY).	○	○	○

Please write any additional comments about DEH web application:

Please write any other ideas you may have about DEH web application that could be essential or desirable:

6. Other initiatives: identification of other initiatives which realized hub as well in order to not start from zero either in terms of framework and technologies or in terms of interesting features.

Initiatives to take as starting point

Please write any already existing initiatives in which DEH can inspire in and propose their interesting features and technologies:

15.2 ***Participants and results collection***

The survey was sent in December, through the online survey-management system (EU survey[11]), to WP3 participants, considering developers as main DEH users. 16 answers were collected.

In the following sections, the collected answers for each section, are shown.

DEH features

To summarise the answers obtained to DEH features closed questions, the following chart shows the aggregated results. From this chart it is possible to conclude that the proposed features were considered by the most as "essential" or "desirable" meaning that were all considered as features to be included in the first version of the DEH.

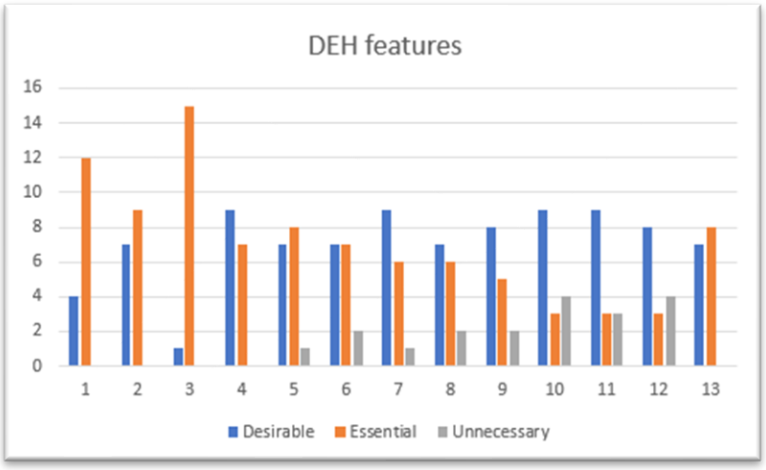


Figure 52: DEH features aggregated answers

Regarding additional features, the following were identified by the participants:

- Provide control Access policies related to their components.
- Easy search and Discovery based on enabler´s objectives.
- DEH to offer suggestions tailored on user's profile.
- Tools that enable the design of a system (collection) of enablers and services to help users (or developers) who fuse together such enablers in order to provide a whole system which can then be sold to other users (e.g. farmers).

[11] https://ec.europa.eu/eusurvey/home/welcome

- Tutorials on usage would be essential; sample data or projects to explore; best practice guidelines.

DEH resources

Regarding the DEH resources, participants did not provide new resource categories with respect to the ones proposed.

DEH interaction

Regarding the DEH Interactions, participants did not provide new resources with respect to the ones proposed.

DEH web application

To summarise the answers obtained to DEH web application closed questions, the following chart shows the aggregated results. From this chart it is possible to conclude that the proposed features were considered by the most as “essential” or “desirable” meaning that were all considered as features to be included in the first version of the DEH.

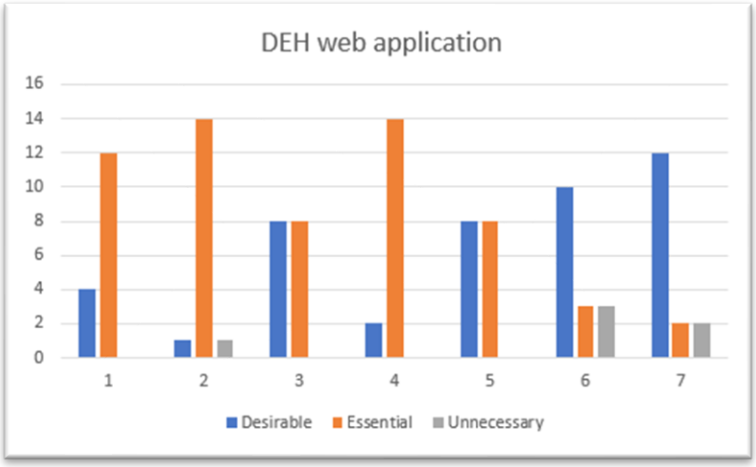


Figure 53: DEH web application aggregated answers

Regarding additional features, the following were identified by the participants:

- the web app should have a page to register solutions and associate to the solution a group of resources, preferable displaying the inter dependencies/relationships between.
- the web app (or some other sort of app) needs to allow users to rate the services or enablers, so the system can be notified e.g. when that service/enabler does not function correctly.
- localisation will be needed, access to open data sources, customisation for each industry/market sector. At a stretch, a team’s feature, so that views can be shared between team members.

DEH Initiatives

Concerning the other initiatives in which DEH can inspire, participants proposed: DatabioHub (https://www.databiohub.eu/registry/), IOF catalogue (https://www.iot-catalogue.com/), Foodie marketplace (https://www.foodie-cloud.org/marketplace/).

16 Appendix D: Component Testing Report Documentation

Table 16 below tabulates the general information of a DEMETER component that is deployed and validated. For more information regarding the validation process please refer to section 11.

Table 16: Component's general description

Title	This field holds the name of the DEMETER component	WP	This field holds the WP that the component belongs
Description	This field holds the component's operation description		
Repository type	This field holds the repository type of the source code of the component. (e.g. Private, DEMETER GitLab)	Justification	This field holds the justification of the source code repository type selection
Repository URL	If the Repository type is in DEMETER's GitLab, then the absolute URL of the component's location must be filled in here		
Integration component list	This field holds the components list that this component interoperates and will integrate with		
Deployment location	This field holds deployment location (e.g. DEMETER cloud infrastructure, DEMETER's pilot infrastructure, proprietary location)		
Docker container location and size	If the component is containerized, then please fill in the location of the docker registry that resides and the size of the docker container		
Requirements	This field holds computational requirements for this component. Among others, you can describe here the CPU, RAM, STORAGE requirements of the component.		
Contact email	This field holds the email of the developer of the component.		

16.1 *<Component> technical description*

In this section, you can give a brief description of the component implementation and operation logic. Also, among others, you can include installation instructions, docker related info or any other helpful information. In most cases you shall elaborate a little bit if necessary, on the information that is described in source code repository. (e.g. README.file etc.)

16.2 *Data Models and Interfaces*

In this section, you can add the Interfaces (APIs) and used Data Models that are described in the component's relevant deliverable.

16.3 *Functionality and Integration Tests*

Integration tests verify that your code works with external dependencies correctly. Unit testing of the component is to be completed for all the features of the described component. Integration tests shall be described and include/adhere the following categories:

1. **Functionality Tests:** Validating the functionality provided by the components to be integrated. Test all the use cases for the components chosen
2. **CRUD operation Tests:** Validating the interconnection between the components to be integrated.
3. **Security Tests:** Validating principles such as data integrity, user authorization where applicable etc.

In this Integration report, each partner shall include the description of the performed integration test for this component. Please also include any useful commends about the integration process. Integration is considered successful when.

- The previous descriptions have been submitted
- Sufficient integration tests have been carried out providing adequate coverage of the functionality provided by each component.